



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁶ : H04Q 11/04, H04L 12/24	A1	(11) International Publication Number: WO 98/47307 (43) International Publication Date: 22 October 1998 (22.10.98)
--	-----------	--

(21) International Application Number: PCT/GB98/00923

(22) International Filing Date: 26 March 1998 (26.03.98)

(30) Priority Data:

9707398.5	11 April 1997 (11.04.97)	GB
08/921,649	27 August 1997 (27.08.97)	US

(71) Applicant (for all designated States except US): NOTHERN TELECOM LIMITED [CA/CA]; World Trade Center of Montreal, 8th floor, 380 St. Antoine Street West, Montreal, Quebec H2Y 3Y4 (CA).

(72) Inventors; and

(75) Inventors/Applicants (for US only): HAYBALL, Clive, Colin [GB/GB]; 9 Bullfields, Sawbridgeworth, Hertfordshire CM21 9DB (GB). BRAGG, Nigel, Lawrence [GB/GB]; Homewards, Chapel Road, Weston Colville, Cambridge CM1 5NX (GB). ROSS, Niall, Forbes [GB/GB]; 2 Blamsters Villas, Little Cambridge, Nr, Great Dunnow, Essex CM6 2DT (GB). TUNNICLIFFE, Andrew [GB/GB]; 1 Waterworks Cottages, Redricks Lane, Sawbridgeworth, Herts CM21 0RL (GB).

(74) Agent: FRANKS, Robert, Benjamin; 352 Omega Court, Cemetery Road, Sheffield S11 8FT (GB).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, GW, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

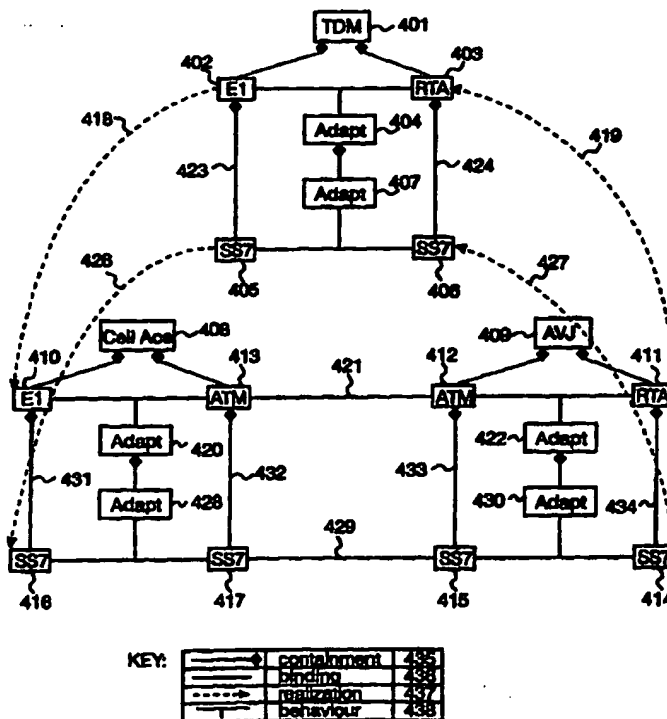
Published

With international search report.

(54) Title: PLANNING SYSTEM FOR BROADBAND MULTI-SERVICE CONNECTIONS

(57) Abstract

A syntactical representation is used to generate data describing a hierarchical broadband network and relationships between its elements. The format of the syntactical representation resembles logic programming language clauses. The syntactical representation is used to specify a broadband connection problem to be solved. Data describing physical resources and functionality of a set of network elements is entered into an inference engine which applies an algorithm to its input in order to find a solution to a specified connections planning problem. The algorithm is recursive and uses artificial intelligence techniques such as first fail, hierarchical planning and hill-climbing to guide the inference engine's search for a solution.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KR	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakhstan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LI	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

-1-

PLANNING SYSTEM FOR BROADBAND MULTI-SERVICE CONNECTIONS

Field of the Invention

5 The present invention relates to communications networks, in particular, though not exclusively to planning of multi-service connections in a hierarchical broadband network.

Background to the Invention

10 Broadband communications networks must deliver a variety of services, often in combination, over transport and access networks which differ geographically in their capabilities and quality of service parameters. Also, communications data is often subject to various forms of transformation, such as compression and translation, as part of its carriage between parties involved.

15

Broadband networks are likely to be heterogeneous for quite some time in the future, in terms of the connectivity types which they offer and the available service gateways. There is therefore a problem when provisioning a service in matching the needs of an end user to the facilities which are available within a
20 service provider.

Planning connections in hierarchical broadband communications networks involves similar planning a route of links and/or nodes in a simple network. However, due to the existence of the hierarchy in the broadband communications
25 network, the routing problem can give rise to a hierarchy of routing sub-problems. A classical routing algorithm (eg Dijkstra's) could be applied to find a route between two nodes. However, the same routing problem could be applied recursively to sub-problems arising in the hierarchical network, but if a sub-problem has no solution then a higher level route must be modified.

30

-2-

Another complication in a heterogeneous broadband network is the differing natures of links between nodes, for example a link between two nodes may only be capable of transferring a certain type of data between the two nodes. Traffic carrying and/or transport characteristics of such nodes constitute "behaviors" of the nodes. Therefore, a corresponding hierarchy of behavior rules need to be found to describe the traffic carrying transport characteristics of individual links and/or nodes in addition to the route of links and/or nodes itself in order to fully plan a connection which provisions a service to the end user.

10 Summary of the Invention

A novel approach outlined herein supports the creation and management of broadband service connections in an integrated manner, using a combination of structured modeling and goal-based route finding.

15 According to a first aspect of the present invention, there is provided in a network comprising a plurality of nodes and links, each said node having at least one end point connecting it to a said link element, each said link between said end points having a type, a method of producing a connection plan representing at least one connection between said end points in said network, said connection
20 plan comprising data describing a set of said end points and said types of said links between the end points said method comprising the steps of:

creating a data representation of at least part of said network;

25 creating a data representation of end points of said connection; and

applying an algorithm to said data representations to generate said connection plan.

-3-

Preferably, a said representation comprises a list of at least one clause, each said clause comprising at least one term.

Said clauses may comprise assertions and implications.

5

Preferably, said algorithm operates to select a said term of a said clause representing said connection as a sub-goal and recursively attempts to solve said sub-goal.

10 A said sub-goal may be assigned a priority, and said algorithm selects a said sub-goal with a highest said priority.

A said algorithm may select a said sub-goal which is a clause with fewest said terms.

15

Said algorithm may select a said sub-goal according to order of its said terms.

20 Said algorithm may select a sub-goal with a lowest position in said network hierarchy.

Said algorithm may select said sub-goal randomly.

25 Said algorithm may select a said clause with which to solve said sub-goal according to position of said clause in said list of clauses.

Said algorithm may select a said clause with which to solve said sub-goal by selecting a said clause having a term included in said connection plan.

-4-

Said algorithm may select a said clause with which to solve said sub-goal by selecting a clause which is lowest in said network hierarchy.

Said algorithm may select a said clause with which to solve said sub-goal
5 by selecting a clause with a term having smallest capacity.

Said algorithm may randomly select a clause with which to solve said sub-goal.

10 Preferably, said clauses represent data describing features of said network, said features selected from the set:

classes of said network nodes;

15 end points of said network nodes;

links of said network;

schematic representations of said link types.

20 A said link type may be selected from the set:

binding;

25 realization;

behavior;

containment.

30

-5-

A graphical representation of said connection plan may be described.

Said connection plan is preferably implemented as a set of connections in said network.

5

Said representation may comprise a substantially Prolog-like syntax.

According to a second aspect of the present invention there is provided in a network comprising a plurality of nodes and links, each said node having at least one end point connecting it to a said link element, each said link between said end points having a type, connection planning apparatus for planning at least one connection between said end points in said network by creating a connection plan comprising a set of said end points and said type of said link between the end point said connection planning apparatus comprising:

15

means of creating a representation of at least part of said network;

means of creating a representation of end points of said connection; and

20 an inference engine which uses said representation to generate said connection plan.

Said representation may comprise a list of at least one clause, each said clause comprising at least one term.

25

Said clauses may comprise horn clauses.

Said inference engine may select a said term of a said clause representing said connection as a sub-goal and recursively attempts to solve said sub-goal.

30

-6-

A said sub-goal may be assigned a priority, and said inference engine selects a said sub-goal with a highest said priority.

Said inference engine may select a said sub-goal which is a clause with
5 said fewest said terms.

Said inference engine may select a sub-goal according to order of its said terms.

10 Said inference engine selects said sub-goal with a lowest position in said network hierarchy.

Said inference engine may select said sub-goal randomly.

15 Said inference engine may select a said clause with which to solve said sub-goal according to position of said clause in said list of clauses.

Said inference engine may select a said clause with which to solve said sub-goal by selecting said clause having a term included in said connection plan.

20

Said inference engine may select a said clause with which to solve said sub-goal by selecting a clause which is lowest in said network hierarchy.

Said inference engine may select a said clause with which to solve sub-goal
25 by selecting a clause with a term having smallest capacity.

Said inference engine may randomly select a clause with which to solve said sub-goal.

-7-

Said clauses may represent data describing features of said network, said features selected from the set:

classes of said network nodes;

5

end points of said network nodes;

links of said network;

10

schematic representation of link types.

A said link type suitably selected from the set:

binding;

15

realization;

behavior;

20

containment.

Preferably, the connection planning apparatus further comprising means for displaying a graphical representation of said connection plan.

25

Said connection planning apparatus means may comprise means for implementing said connection plan as a set of connections in said network.

According to a third aspect of the present invention, there is provided a syntactical representation for describing capabilities and links in a hierarchical
30 network, said syntactical representation comprising a list of at least one clause.

Said capability representation preferably comprises:

a node identifier;

an input data type;

an output data type; and

a hierarchical network layer upon which said node appears.

Said link representation preferably comprises:

at least two nodes representing end points of said links; and

a hierarchical network layer upon which said link appears.

Said representation may comprise a substantially Prolog-like syntax.

According to a fourth aspect of the present invention there is provided an inference engine for solving connection problems in a hierarchical network comprising:

a logic component comprising a specification of what it means to solve said problem; and

a control component comprising a description of how said logic component is to be executed.

-9-

Said control component preferably comprises:

a computation rule for selecting sub-goals with which to solve said problems; and

5

a selection rule for selecting clauses with which to solve aid selected sub-goals.

10 An inference engine may execute said logic component by applying techniques selected from the set:

hierarchical planning;

hill-climbing;

15

first fail.

20 According to a further aspect of the present invention there is provided a network management apparatus for a communications network comprising a plurality of interconnected network element devices, said network management apparatus comprising:

25 a plurality of device controllers each controlling at least one corresponding network element device, said plurality of device controllers comprising a plurality of device sub-controllers,

wherein a plurality of functions performed by a network element device are each represented by a corresponding respective said device sub-controller; and

-10-

said plurality of device controllers may be arranged in a manner which represents an arrangement of a service provided by said network element devices.

5 Preferably each said device sub-controller comprises: a data storage means storing data describing a function of a network element device; and a means for generating management control signals for controlling a said network element device. The device sub-controller preferably comprises an object data representation.

10

Preferably a set of connections between said device controllers are provided, said connections arranged in a manner which represents service connections between said network element devices.

15 This invention includes in a network management apparatus, a method of controlling a network comprising a plurality of network element devices each supporting a plurality of services implemented by functions of the network element devices, said method comprising the steps of:

20 representing each said function supported by a network element device by a corresponding respective device sub-controller by;

at each said sub-controller storing data signals representing a said function capability of said network element device; and

25

operating said sub-controllers to generate a plurality of management control signals for controlling said network element device to provide said functionality in supporting said services.

-11-

The invention includes in a network management apparatus, a method of controlling a network comprising a plurality of network element devices each supporting a plurality of service functions, said method comprising the steps of:

5 representing each network element device by a corresponding device controller;

 for each device controller, representing each function capability provided by a said corresponding network element device by a corresponding device sub-
10 controller; and

 linking a plurality of said sub-controllers of different device controllers in a manner which represents a service connection supported by said network element devices.

15

The invention includes a method of constructing a management apparatus for a communications network comprising a plurality of node element devices, said method comprising the steps of:

20 representing a plurality of network element devices of the network by a corresponding plurality of function models, wherein each function model comprises a plurality of specifications of functionality of a said corresponding network element device; and

25 implementing said function models as a plurality of device controllers each controlling a corresponding respective said network element device, wherein each said device controller comprises a plurality of device sub-controllers each device sub-controller corresponding to a service function provided by a network element device controlled by said device controller.

30

-12-

Preferably each said device sub-controller comprises: a data storage medium for storing a set of data signals describing a functionality of a network element device; and a control signal generating means for generating control signals for controlling a function of said network element device.

5

The invention includes in a communications network comprising a plurality of inter-connected nodes, each node providing a plurality of services, a method of planning a service between a source node and at least one destination node comprising the steps of:

10

storing data representing connections between said nodes;

storing data representing services provided by said nodes;

15

finding data representing a set of connections and services between said source and destination nodes which implement said service; and

transforming said data representing a set of connections and services for application within a network controller.

20

A said service provided by a node may comprise transforming a first service type to a second service type.

Said data representing connections may comprise a name of a first node, a
25 name of a second node and a name of a layer within which said nodes are contained.

Said data representing services may comprise a name of a node, a name of a first service type provided by said node and a name of a second service type
30 provided by said node.

-13-

Said data transformed for application within a network controller may comprise an aggregated connection between said network nodes for implementing a said service.

5

The invention includes in a communications network comprising a plurality of inter-connected nodes, said nodes supporting at least one service function, a method of planning a connection supporting said service, said method comprising the steps of:

10

translating function capabilities of said nodes and connections between said nodes into a plurality of syntax constructs;

taking each said syntax construct as an operator in a planning system to
15 obtain a connection plan solution; and

transforming said connection plan solution into a connection model suitable for application within a network manager apparatus.

20 A said syntactical operator may comprise a connection operator.

A said syntactical operator may comprise a transform operator representing a said function capability of a said node.

25 A said service may be represented in terms of a layered structure.

Said connection plan solution is preferably obtained using an artificial intelligence technique.

30 A said connection may comprise a multi-media service connection.

Brief Description of the Drawings

For a better understanding of the invention and to show how the same may be carried into effect, there will now be described by way of example only, specific embodiments, methods and processes according to the present invention with reference to the accompanying drawings in which:

Fig. 1 herein illustrates schematically a portion of a broadband network (in practice, a broadband network would mainly have sub-layers, nodes and connections);

Fig. 2 herein illustrates a model for an architecture of a network management system for managing network elements of the network of Fig. 1 herein;

Fig. 3 herein illustrates an arrangement of a network controller apparatus for controlling individual network elements across the network of Fig. 1 herein;

Fig. 4 illustrates schematically an Application Model representing a hierarchical network comprising instances and links between said instances;

Fig. 5 illustrates schematically an algorithm according to the preferred embodiment comprising a create representation step, a specify solution step and a search for solution step;

Fig. 6 illustrates schematically steps executed during the search for solution step identified in Fig. 5, including a select sub-goal step and a decide how to solve selected sub-goal step;

-15-

Fig. 7 illustrates schematically steps performed during the select sub-goal to solve step of Fig. 6, including a select sub-goal according to mode and priority steps and a link sub-goal selection step;

5 Fig. 8 illustrates priorities and modes associated with sub-goals used by the steps identified in Fig. 7;

Fig. 9 illustrates steps performed during the link sub-goal selection step of Fig. 7;

10

Fig. 10 illustrates steps performed during the decide how to solve selected sub-goal step of Fig. 6.

Fig. 11 illustrates a simplified Application Model;

15

Fig. 12 illustrates schematically steps which may be performed by the preferred embodiment whilst solving a connection problem relating to the Application Model of Fig. 11;

20 Fig. 13 illustrates execution of the steps of Fig. 12 on the Application Model of Fig. 11;

Fig. 14 illustrates further steps which may be executed by the preferred embodiment on the Application Model of Fig. 11; and

25

Fig. 15 illustrates execution of the steps of Fig. 14 on the Application Model of Fig. 11.

Detailed Description of the Best Mode for Carrying Out the Invention

-16-

There will now be described by way of example the best mode contemplated by the inventors for carrying out the invention. In the following description numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent however, to one skilled in the art, that the present invention may be practiced without using these specific details. In other instances, well known methods and structures have not been described in detail so as not to unnecessarily obscure the present invention.

Referring to Fig. 1 herein, there is illustrated schematically a portion of broadband communications network comprising a plurality of nodes and links in which there exists a connection planning problem described with reference to a hybrid network including asynchronous transfer mode (ATM) and T1 transmission modes. An end user apparatus 100 eg a telephone can only communicate by voice signal, being connected into a public switched telephone network (PSTN) at a local exchange. Across on another side of ATM network 101 is a text generator system 102. Somewhere within the scope of the ATM network 101 is a text to voice converter 103 for converting text generated by the text generator to voice signals. Access between T1 network 104 and ATM network 101 is made via a T1-ATM access apparatus 105.

20

The problem in this example is that a user of the end user apparatus 100 wishes to receive a voice message from text generated by the text generator 102. The text generator 102 generates text data signals which are converted to voice signals by the text to voice converter 103, which are then relayed across the ATM network and T1 network via T1-ATM access node 105 to the end user. An operator at the text generator 102 enters text which is received as a voice signal by the end user 100 over their telephone apparatus.

This is one example of a multimedia service connection problem. In the context of general hybridized multimedia service supporting networks, there may

30

-17-

be a large number of problems of this type, eg text to voice conversion, voice to text conversion, data conversion etc. These problems are potentially highly complex to solve, and complex to manage using a network manager system.

5 A problem is that not only must one be able to work out how to establish the necessary connectivity, to achieve an end-to-end service, but one must be able to map that connectivity onto the network management means to manage the physical resource hardware providing that service.

10 In the proposed approach herein, the connectivity and interworking capabilities offered by a broadband communications network are modeled in terms of:

- 15 (i) various transport and service sub-layers, nodes and connections between nodes;
- (ii) cross-connect and interworking functions supported within nodes.

 The capabilities illustrated in this example might be used to support a
20 variety of different user services, including the insertion of automatically generated voice fragments from a library of text strings.

 Figs. 2 and 3 herein illustrate one specific example of how the network element devices shown in Fig. 1 may be modeled, based on notations used
25 within the synchronous digital hierarchy (SDH) and ATM management domains.

 The notation of the model shown in Fig. 2 herein is based upon an application model concept as disclosed in the applicant's previous disclosure US 08/921,649 filed 27 August 1997, a copy of which is filed herewith, and the
30 contents of which are incorporated herein by reference.

-18-

An application model comprises a representation of a plurality of application level elements, being elements of a management system relating to a physical resource such as a component, composite or system in which a functionality of said component, composite or system is represented independently of an internal structure of said component, composite or system.

An application level element may be capable of providing management signals relating to all aspects of managing a functionality of a component, composite or system independently of internal structure, for example providing signals describing a status or performance of a component, composite or system in terms of functionality independently of its internal structure or providing control of a component, composite or system independently of its internal structure. Suitably, application level elements are constructed from a set of managed objects, for example in accordance with the ITU-T TMN architecture standard.

An application model comprises a model for an external interface, ie a "black box" view of modeled physical resources, which can be used by a human operator or an application program.

20

An implementation model comprises representation of a collection of implementation level elements, which represent and interface with physical resources, and in which the specific internal layout of those resources, ie the physical and logical implementation details of physical resources are represented.

25

Each node of the network, comprising a physical network element device such as the text generator 102, text to voice converter 103, T1-ATM access node 105 and end user apparatus 100, is represented by a respective device controller 300, 301, 302, 303 in a network controller apparatus, operating to control node

30

-19-

element devices of the network via operation, administration and maintenance (OAM) control signals transmitted across the network. The device controllers may comprise managed object as described in the ITU-T TMN architecture specifications. The device controllers 300-303 are structured and interconnected
5 so as to represent the service functionality provided by their corresponding respective network element devices 102, 103, 105, 106. For example a text generator device controller 200 supports an ATM connection termination point function and a text connection termination point function. Similarly, a text to voice converter device controller 301 supports a pair of ATM connection termination
10 point functions 204, 205, a text connection termination point function 206 and a voice connection termination point function 202, and text to voice conversion function 208.

Each device controller 300-303 comprises a plurality of sub-controllers,
15 each of which controls a function provided by the corresponding network element device. The sub-controller may comprise managed objects as described in the ITU-T architecture specifications. For example the text generator 102 provides an ATM connection termination point facility which is controlled by ATM connection termination point sub-controller 309, and text generator 102 provides
20 a text connection termination point which is supported by text connection termination point function sub-controller 310. Each sub-controller is implemented as a data storage device containing data signals describing functionality, eg functions provided by a device, and a means for generating function control signals for controlling that functionality. The sub-controllers of different device
25 controllers in the network manager apparatus are interconnected through signal channels, representing service connection, links, paths and service functions.

In order to construct the network controller comprising the plurality of device controllers and constituent sub-controllers as shown in Fig. 3, firstly the network

-20-

is modeled, resulting in a function model data which specifies functionality and connectivity, as shown schematically in Fig. 2 herein.

In Fig. 2 herein, each network element device is modeled in terms of the functions which it provides, and in Fig. 3 herein the network controller is constructed as a set of device controllers and sub-controllers which mirrors the functional structure of the network element devices themselves. For example a functional model 200 is prepared describing the text generator device 102. The functional model comprises a model 210 for an ATM connection termination point and a model 211 for a text connection termination point. The functional models are implemented in device controller 300 comprising part of a network controller apparatus, as an ATM connection termination point sub-controller 309 and a text connection termination point sub-controller 310.

Connections between the sub-controllers, embedded as part of the control signals sent to other sub-controllers are used to operate the corresponding network elements to provide connections, links, virtual paths and other services across the network. By reconfiguring the connections between sub-controllers, in the device controllers, the network can be managed as this interconnectivity is reflected in the actual network element devices which receive the OAM control signals from the network controller to implement these reconfigurations to the functionality of the corresponding network element devices.

The model may be formalized by translating the capabilities of each node and the connections between nodes into a syntax suitable for route finding.

The basic elements of the syntax are:

connection (S, X, Y)

30

-21-

(meaning a layer S connection between locations X and Y)

and:

5 *transform (S, T, X)*

(meaning an interworking function between service types S and T at node X

- reduced to a cross-connect function when $S=T$)

10 In each of the above cases, the service type may itself be expressed in terms of layered structure, eg $S = \text{Voice/ATM}$ (meaning the service is a voice over ATM service).

The connection problem is now solved in terms of a planning task which
15 takes one end of the desired service connection as its start state, the other end as its goal state, and each connection and/or transform construct as an operator. Further constraints may be added to represent the quality of service parameters or other characteristics of network components. A known artificial intelligence technique may thus be applied to solving a multimedia service connection
20 problem by finding an optimized, or at least a workable, solution of connections and transforms using the links and nodes as components of the solution.

Once a viable solution has been found, the resultant plan may be transformed back into a layered connection model suitable for application within a
25 network management environment based on a network controller apparatus.

There now follows an application example for modeling the capabilities of the network example of Fig. 1. The capabilities represented in Fig. 1 may be represented syntactically as follows:

30

-22-

connection (text / ATM, A, B).

(meaning ATM text connection between nodes A and B, ie text data may be transferred between locations A and B)

5

transform (text / ATM, voice / ATM, B).

(meaning node B is capable of converting ATM text data to ATM voice data)

10

connection (voice / ATM, B, C).

(meaning an ATM voice connection between nodes B and C, ie voice data may be transferred between locations B and C)

15

transform (voice / ATM, voice / T1, C).

(meaning node C is capable of converting ATM voice data to T1 voice data)

connection (voice / T1, C, D).

20

(meaning T1 voice connection between nodes C and D, ie voice data may be transferred between locations C and D)

The planning goal is to connect text at node A to voice at node D. The solution (unique in this example) is a connection A-B-C-D with the text-voice conversion being carried out at node at B and voice cross-connection carried out at node C. The solution can be superimposed back onto the network model in the form of an aggregated connection 311 as shown in Fig. 3 herein, the aggregated connection 311 may be represented syntactically as follows:

30

-23-

*connection (text/ATM, A, B), transform (text/ATM, voice/ATM, B),
connection (voice/ATM, B, C), transform (voice/ATM, voice/T1, C), connection
(voice/T1, C, D)*

- 5 There will now be described possible extensions and other applications of
the methods disclosed herein. The connection planning technique described
above could be greatly facilitated by the pre-discovery of the "physical"
connection topology of each connectivity service layer. For example, ATM
networks will deploy PNNI (Private Network Node Interface) technology to allow
10 ATM switching and end-user nodes to discover an abstract view of the ATM
network topology (via peer-to-peer interaction) and to route connections across
the discovered topology via selected routing algorithms. This concept could be
applied to higher level connectivity services, starting perhaps with ATM
adaptation layers, especially AAL2. The connections available at each
15 connectivity service layer can be viewed in terms of Virtual Private Networks
(VPNs) constructed across the next lower connection layer.

- Another example of configuring a network manager using the connection
planning method is the provisioning and configuration of distributed network
20 elements. Configuration and provisioning of distributed network elements, utilizes
the same technique as described hereinabove, but applied at "microcosmic" level
of connections within nodes. An example would be the configuration of a
signaling link between an access port and an internal frame processing device.
This technique, coupled with a design tool to support the constructions system
25 management models from component fragments, may make a major impact
upon the time and effort involved in management system development.

- A further formalization of the syntactical representation according to the
preferred method will now be described, as well as algorithms including AI
30 algorithms preferably executing on a network controller workstation which may be

-24-

applied to data created according to the syntactical representation in order to implement it in solving problems such as broadband connections planning.

A preferred embodiment and method according to the present invention will
5 be illustrated with examples, the first of which is shown in Fig. 4 of the accompanying drawings.

In the preferred embodiment and method, data describing at least part of the hierarchical broadband network and a connection problem to be solved is
10 described using a syntactical representation as disclosed herein above. An algorithm is applied to the network and problem data which attempts to solve the problem by generating a connection plan comprising a set of network nodes and/or links or other data describing network resources/elements which may be allocated to a connection in order to implement it in the network. The inventors
15 envisage that running the algorithm to generate such a connection plan may be used to select a set of network connections to be implemented which can best satisfy certain criteria, eg balanced usage of network resources, etc.

The examples described herein use notation and concepts described in the
20 applicant's co-pending application US 08/921,649 of 27 August 1997. The example shown in Fig. 4 comprises three classes: TDM, AVJ and Cell Ace with one instance of each class, labeled 401, 406 and 409 respectively. Fig. 4 comprises an Application Model (as disclosed in the patent applications referenced hereinabove) of the three instances. An application model comprises
25 a set of objects which represent functionalities provided by real physical resources, without describing the specific implementation details of those underlying physical resources.

Fig. 4 illustrates relationships between the application model's components.
30 The relationships are referred to herein as "links". Four types of link may exist:

-25-

- Realizations. Herein, by the term realization, it is meant a relationship between an application model and an implementation model.
- 5 • Behaviors. Herein, by the term behavior, it is meant a class which models a data transforming behavior in a physical resource, eg a network device, or a class in an application model which models a function represented by the application model.
- 10 • Containment. Herein, by the term containment, it is meant a subordinate element (eg a managed object) of an application model, which is subordinate to a superior element (managed object) of the application model. An subordinate element which is contained by a superior element may be accessible through the superior element.
- 15 • Bindings, eg, A data transfer link between the application model's components, which may be implemented by means of a physical link.

In a broadband connection planning problem, a solution to the problem may
20 comprise a set of links and/or nodes forming a route between two nodes. A graph such as that shown in Fig. 4 may describe an Implementation Model detailing which nodes and/or links of the Application Model may be used to implement the connection. Possible routes of links existing between nodes shall be referred to herein as "physical links" and routes comprising a set of physical
25 links selected during execution or as a result of executing the algorithm are called "logical links". Containment links 435 may be illustrated as lines between network nodes having a diamond shape at one end, denoting that a network node at the diamond end of the line contains a network node on another end of the line. A binding link between nodes may be illustrated by straight lines 436. A realization
30 link 437 may be illustrated by dotted lines having an arrow at one end denoting

-26-

that a node at the arrow end of the dotted line realizes a node at another end of the line. A behavior link 438 existing between two nodes may be illustrated by a horizontal line, each end of which is connected to one of the nodes and a vertical line near a center of the horizontal line is attached to a box indicating a nature of the behavior link. Links usually occur between end points of instances, the end points possibly representing ports on a switch for example.

A "port" is defined in recommendation G.805 as "a pair of uni-directional ports". A uni-directional port as defined in recommendation G.803 "represents the output of a trail termination source of a uni-directional link connection, or the input to a trail termination sink or uni-directional link connection". Logical ports exist within a network element and may have similar layer structures and characteristics as physical ports, but do not actually bind to any physical port externally to the network element. Logical ports may be bound to one card, or may float across a plurality of cards. The physical or logical ports of a card may all be of a same type, or several different types of port may be resident on a card, depending on the specific manufacture of the network element itself.

Specific physical or logical ports are represented in the specific implementation herein as end points. In this specification, the term "end point" is used to describe a port comprising a receive port, and (optionally) a transmit port.

In Fig. 4 TDM instance 401 appears at a top level of a class-level hierarchy and contains an E1 end point 402 and an RTA end point 403. End points 402 and 403 are linked by an adapt behavior 404. End point 403 contains (423) an SS7 end point 407 and RTA end point 403 contains (424) an SS7 end point 406. End points 405 and 406 are linked by an adapt behavior 407, which is contained by adapt behavior 404.

-27-

Cell Ace instance 408 appears at a lower level in the class-level hierarchy than TDM instance 401 and contains an E1 end point 410 and an ATM end point 413. End points 410 and 413 are linked by an adapt behavior 420. E1 end point 410 contains (431) an SS7 end point 416. ATM end point 413 contains (432) an
5 SS7 end point 417. End points 416 and 417 are linked by an adapt behavior 428, which is contained by adapt behavior 420.

AVJ instance 409 appears at the same level of the class-level hierarchy as Cell ace instance 408 and contains an ATM end point 412 and an RTA end point
10 411. End points 412 and 411 are linked by adapt behavior 422. ATM end point 412 contains (433) an SS7 end point 415. RTA end point 411 contains (434) an SS7 end point 414. End points 414 and 415 are linked by an adapt behavior 430, which is contained by adapt behavior 422.

15 E1 end point 402 of TDM instance 401 is realized (418) by E1 end point 410 of Cell Ace instance 408. RTA end point 411 of AVJ instance 409 is realized (419) by RTA end point 403 of TDM instance 401. SS7 end point 405 of TDM instance 401 is realized (426) by SS7 end point 416 of cell ace instance 408. SS7 end point 414 of AVJ instance 409 is realized (427) by SS7 end point 406 of
20 TDM instance 401. ATM end point 413 of Cell Ace instance 408 is linked by binding 421 to ATM end point 412 of AVJ instance 409. SS7 end point 417 of Cell Ace instance 408 is linked by binding 429 to SS7 end point 415 of AVJ instance 409.

25 Data describing the application model illustrated in Fig. 4 is preferably used as an input to the algorithm. As well as data describing the classes, instances and links relating to the Application Model, the input data may also comprise behavior rules and optional partial implementation rules for architecture instances which comprise pre-defined logical links. Additional input data may also be used,
30 such as: end point types and capacities; protocol expansions; physical link

-28-

capacities and end point type realization rules. Optionally, partial Implementation Models consisting of pre-defined logical links for architecture class instances may also be accepted as inputs to the algorithm. In the example of Fig. 4, the input in partial implementation model is trivial, ie it contains no pre-defined logical links.

5

Additionally, input to the algorithm should comprise a specified Application Model instance that is to be provisioned (typically at a top of the class level hierarchy), ie the connection problem to be solved. In the example given in Fig. 4, the specified instance to be provisioned comprises TDM instance 401. The TDM instance's E1 end point 402 and RTA end point 403 and the adapt behavior 404 between them must be provisioned, and also the two SS7 end points 405 and 406 and the adapt behavior 407 between them. The end points are preferably provisioned by realizing them, ie constructing a sequence of behaviors between the realizing end points and so on as described hereinbelow. In the example, the sequence of behaviors constructed may be:

adapt o adapt

corresponding to behavior rule:

20

adapt = adapt o adapt

The Cell Ace instance 408 and AVJ instance 409 have equipment models and do not require provisioning.

25

Fig. 5 of the accompanying drawings illustrates steps typically executed by the algorithm when attempting to solve a broadband connections planning problem. At step 501 a representation of the problem is created, preferably the problem representation comprises the syntactical representation as described herein and is stored in the network controller's data storage. Data comprising a

30

-29-

list of one or more clauses may be entered into an inference engine. After the algorithm has created the representation the inference engine is applied to the representation in order to attempt to find a solution to the problem. An inference engine may comprise rules or an algorithm intended to solve a given problem, preferably by producing inferences based on facts and rules, such as those described by the syntactic representation. The inference engine preferably comprises two parts: a logic component and a control component. At step 502 the inference engine's logic component specifies what it means to solve the problem, ie defining how to provision an instance. At step 503 the inference engine's control component describes how the logic component is to be executed, ie how it will be used to search for the specified solution.

The representation of a broadband connections planning problem created at step 501 may comprise the following data:

15

- classes, each declared as either equipment or architecture (corresponding to equipment and architecture models as described in the patent applications referenced hereinabove) and placed in the class-level hierarchy

20

- end points with unique identifiers

- end point types and capacities

25

- physical links (defined explicitly or implicitly)
- logical links selected from the physical links

-30-

- behavior rules of the form $b = b_1 \circ \dots \circ b_N$ wherein the \circ operator denotes composition of behaviors and corresponds to a binding between behavior end points. Families of behavior rules may be defined implicitly

5 In the preferred embodiment, models, physical links, behavior rules and other information describing an Application Model and a connection planning problem to be solved by the algorithm may be represented textually as clauses. Clauses, in particular Horn clauses, form the basis of Prolog, a known logic programming language (see, for example, Ivan Bratko, "Prolog: Programming for
10 Artificial Intelligence" second edition, Addison Wesley, 1990). The preferred embodiment's syntactical representation comprises a substantially Prolog-like syntax (preferably with convenient extensions such as $S=\{t(X)|p(X)\}$ for set construction). The syntactical representation is not necessarily a Prolog program; it preferably comprises data describing a formalism to which the algorithm's
15 inference engine, as described hereinbelow will be applied.

Clauses may be one of three types: facts, rules and questions. Each clause usually terminates with a full stop (.). Facts declare things that are unconditionally true, for example:

20

network (atm)

declares that an object *atm* is always a *network* object. Each clause may comprise at least one object called a "term". In the example above, *atm*
25 comprises a term. A term beginning with a lower case letter may be considered to be a constant (or an "atom"). Terms beginning with capital letters may be considered to be variables. A term inside a clause's brackets may be called an "argument" and a term prefixing the brackets a "functor".

-31-

A rule preferably comprises a condition part (on a right hand side of a :- symbol) and a conclusion part (on the left hand side of the :- symbol). The conclusion part may be called a head of the rule clause and the condition part may be called a body of the rule clause. If the condition part is true then a logical
5 consequence of this is that the conclusion part is true.

A question may comprise a clause having a variable which is unbound, ie has not been defined as having a specified value. A question may be answered by using an inference engine algorithm which utilizes facts and rules defined to
10 substitute the variable with another object. In a question comprising one or more unbound variable, each unbound variable may be selected as a goal, therefore all goals within a question should be satisfied. To satisfy a goal means to demonstrate that the goal logically follows from facts and rules defined.

15 Variable substitution primarily involves term matching. Two terms match if they are identical or if variables in both terms can be substituted with objects in such a way that after the substitution of variables the terms become identical. Conventionally in Prolog matching of goals occurs in a left to right order of terms appearing in a clause. The preferred embodiment uses a more sophisticated
20 goal selection process as described hereinbelow.

The syntactical representation may specify a class by the following clause:

class_info (Class, Type, Level)

25

wherein *Type* may be either *equipment* or *architecture*, indicating that the class belongs to the implementation model or application model respectively; *Class* comprises an alpha-numeric identifier for the class and *Level* comprises an integer denoting the class's position in the class-level hierarchy (0 denoting a

-32-

highest level usually). Thus, the *class_info* clauses for the example shown in Fig. 4 may be:

5 *class_info* (*TDM, architecture, 0*) representing the TDM class of the
Application Model

class_info (*Cell_Ace, equipment, 1*) representing the Cell Ace class

10 *class_info* (*AVJ, equipment, 1*) representing the AVJ class

The syntactical representation's clause for an end point preferably comprises:

15 *endpoint* (*Class, Instance, Id*)

wherein *Class* to which the end point being representing by the clause belongs; *Instance* may comprise an identifier term for the end point's class instance and *Id* may be an integer which is unique to the class, making the *endpoint* clause a unique identifier for each end point in the problem.
20 Alternatively, *Id* may be an actual identifier generated by the network controller.

The syntactical representation may also include capacity data for end points, for consistency checking of bindings or for behavior rule and physical link schemata. The end point and capacity data may be represented by the following
25 clause:

class_endpoint (*Class, Id, Type, Capacity*)

wherein *Class*, *Id* and *Type* may be as in an *endpoint* clause and *Capacity*
30 may be an integer describing a capacity of the end point being represented by

-33-

the clause, for example representing bandwidth capacity. The algorithm may take into account capacity usage of end points by other connections when computing a connection plan for the specified instance.

5 Thus, for the problem illustrated in Fig. 4 the *class_endpoint* clauses may be:

```

class_endpoint( tdm, 1, e1, 0)      for end point 402
class_endpoint( tdm, 2, rta, 0)     for end point 403
10  class_endpoint( tdm, 3, ss7, 0)  for end point 405
class_endpoint( tdm, 4, ss7, 0)     for end point 406
class_endpoint( cell_ace, 1, e1, 0) for end point 410
class_endpoint( cell_ace, 2, atm, 0) for end point 413
class_endpoint( cell_ace, 3, ss7, 0) for end point 416
15  class_endpoint( cell_ace, 4, ss7, 0) for end point 417
class_endpoint( avj, 1, rta, 0)     for end point 412
class_endpoint( avj, 2, rta, 0)     for end point 411
class_endpoint( avj, 3, rta, 0)     for end point 415
class_endpoint( avj, 4, rta, 0)     for end point 414

```

20

End point types may be used by the algorithm to check for consistency of end point containment rules. End point types may be used in a physical link schemata, for example for implicitly specifying behaviors between all end points of a same time in a given class.

25

An instance's end point in the Application Model may be given in the syntactical representation by the following clause:

```
t (Class, Instance, Id)
```

30

-34-

wherein *Class* is preferably the instance's class; *Instance* is the class instance which the end point represented by the clause occurs and *Id* represents the identifier of the end point.

5 The syntactical representation's clause for a physical link may be as follows:

link (T1, T2, Behavior)

10 wherein *Behavior* may be one of *realize*, *bind*, *contain* or a specific behavior prefixed by a hyphen (-) representing a type of the link, ie ITS nature, that is a capability or function, of the link between end points *T1* and *T2*. Thus, for the example illustrated in Fig. 4, the *link* clauses may comprise:

15	<i>link(t (tdm, 1, 1), t (tdm, 1, 2), -adapt)</i>	for adapt behavior 404 between end points 402 and 403
	<i>link(t (tdm, 1, 3), t (tdm, 1, 4), -adapt)</i>	for adapt behavior 407 between end points 405 and 406
20	<i>link(t (tdm, 1, 1), t (cell_ace, 1, 1), realize)</i>	for realize link 418 between end points 402 and 410
	<i>link(t (tdm, 1, 2), t (avj, 1, 2), realize)</i>	for realize link 419 between end points 403 and 411
25	<i>link(t (avj, 1, 1), t (avj, 1, 2), -adapt)</i>	for adapt behavior 422 between end points 412 and 411

-35-

	<i>link(t (avj, 1, 3), t (avj 1, 4), -adapt)</i>	for adapt behavior 430 between end points 415 and 414
5	<i>link(t (cell_ace, 1, 1), t (cell_ace, 1, 2), -adapt)</i>	for adapt behavior 420 between end points 410 and 413
	<i>link(t (cell_ace, 1, 3), t (cell_ace, 1, 4), -adapt)</i>	for adapt behavior 428 between end points 416 and 417
10	<i>link(t (cell_ace, 1, 2), t (avj, 1, 1), bind)</i>	for bind link 421 between end points 413 and 412
	<i>link(t (tdm, N, 1), t (tdm, N, 3), contain)</i>	for contain link 423 between end points 402 and 405
15	<i>link(t (tdm, N, 2), t (tdm, N, 4), contain)</i>	for contain link 424 between end points 403 and 406
	<i>link(t (cell_ace, N, 1), t (cell_ace, N, 3), contain)</i>	for contain link 431 between end points 410 and 416
20	<i>link(t (cell_ace, N, 2), t (cell_ace, N, 4), contain)</i>	for contain link 432 between end points 413 and 417
	<i>link(t (avj, N, 1), t (avj, N, 3), contain)</i>	for contain link 433 between end points 412 and 415
25	<i>link(t (avj, N, 2), t (avj, N, 4), contain)</i>	for contain link 434 between end points 411 and 414

30

-36-

Symmetric rules (which include most binding and behavior links) require two clauses. For example:

```

link (t (tdm, 1, 1), t (tdm, 1, 2), -adapt
5  link (t (tdm, 1, 2), t (tdm, 1, 1), -adapt

```

For broadband connections planning problems of a realistic size it may be impractical to explicitly enter data describing all physical links. Therefore, physical links may be defined implicitly: for example using a physical link schema
 10 for physical link behaviors in an instance with an unlimited number of end points, any two of which may be linked by a trivial behavior (which has no effect upon data carried):

```

link(t (atm_switch, N, A), t (atm_switch, N, B),-trivial):- integer(A), integer(B)
15

```

wherein *integer* generates an unbounded series of integers. Alternatively, instead of *integer* a predicate may be used which interrogates a database stored by the network controller in order to obtain an end point identifier. Other possible uses may include:

20

- Handling realizations between contained end points: if *T1* physically realizes *T2*, *T1* contains *T3* and *T2* contains *T4*, then *T3* physically realizes *T4*. Adding a rule describing this realization and containment links to a *link clause* may be achieved by only explicitly specifying
 25 realizations between end points at tops of containment links.

- Grouping sets of instances by location, allowing physical links between any two end points with matching types at the same location.

30

- Specifying only physical links between equipment end points.

-37-

- Physical links between architecture end points may be implicit.

A logical link may be described by the syntactical representation by the
 5 following clause:

*logical_link (T1, T2, Links, IM):- link (T1, T2, Link), add_logical_link (T1, T1,
 Link, IM)*

10 wherein the *link* term may represent a physical link selected for inclusion in
 the logical link being represented by the clause (which may be considered as a
 first/head link in a list of links); *add_logical_link* may be a term for adding
 subsequent links to the list which comprise the logical links and *IM* may be a term
 containing a variable for each possible pair of end points and each variable
 15 becomes bound when the two end points are assigned a logical link.

logical_link comprises a record of created logical links which is maintained
 during execution of the algorithm. The record is preferably backtrackable
 because logical links are removed on backtracking. When adding a logical link,
 20 other logical links must often be forbidden. For example, it is incorrect to select
 logical behaviors between end points T1 and T2 and between T3 and T4, where
 T1 contains T3 and T4 contains T2. This relationship may be referred to as "No
 Klein Bottle Rule" and may be implemented by binding appropriate *IM* variables
 to a special term *forbidden*. As variable bindings are used for forbidding logical
 25 links, these correctly become re-enabled upon backtracking.

A consequence of using variable bindings to represent logical links is that
 no two end points may be linked by more than one kind of logical link
 simultaneously, though on backtracking different links may be assigned. Another
 30 consequence is that the algorithm can "create" the same logical link more than

-38-

once without ill effect. This property allows a partial Implementation Model to be specified as part of the broadband connections planning problem input, simply by creating predefined logical links.

- 5 The syntactical representation may describe behavior rules using the following clause:

behavior_rule (b,[b1,...,bN])

- 10 which may represent the behavior $b = b_1 \circ \dots \circ b_N$ wherein the \circ operator denotes composition of behaviors, and corresponds to a binding between end points. The *trivial* behavior may be defined as:

$b = \text{trivial} \circ b = b \circ \text{trivial}$

15

The *behavior_rule* clause may be generalized to include operators corresponding to links other than bindings. The behaviors $b_1 \dots b_N$ may also be compound terms containing parameters as arguments.

- 20 Families of behavior rules may be described by the syntactical representation by clauses such as the following:

behaviour_rule (B, [B|Bs]): -behavior_rule_generator(B, Bs).

behaviour_rule (B, [trivial|Bs]): B = trivial behavior_rule (B, Bs)

25

behaviour_rule_generator (B, []): -behavior_rule_generator (B, [B|Bs])

These clauses may generate an unbounded number of behavioral rules such as:

30

adapt = adapt

-39-

adapt = adapt o adapt

adapt = adapt o trivial o adapt ...

These clauses may be generalized to handle other operators and rules.

5

The input to the algorithm also includes the specific instance to be provisioned, ie a goal of the broadband connections planning problem. The syntactical representation of the problem's goal may be:

10 $\text{:- provision_instance } (T1, T2, B, IM) .$

wherein T1 and T2 are bound to end points on the same instance: *B* to a behavior and *IM* is unbound. Solving the goal binds *IM* to a representation of the implementation model.

15

The syntactical representation of the specific instance to be provisioned in the example of Fig. 4, ie end points 1 and 2 of *tdm* instance 401, may be:

$\text{:- provision_instance } (t(tdm, 1, 1), t(tdm, 1, 2), adapt, IM)$

20

Once the syntactical representation of the problem has been created and input into the algorithm, a syntactical representation specifying what it means to solve the problem may be created by the algorithm at step 502 by the inference engine's logic component. The specification is preferably a recursive description of what it means to provision an instance, specifically two of its end points and a behavior link between them.

25

In order to solve the broadband connection planning problem, the specified instance must be provisioned down to equipment instances. Rules for provisioning are preferably:

30

-40-

- To provision an instance: given two end points on an application model instance and a logical behavior between them:

5 if the instance comprises *equipment* then the instance is provisioned

 otherwise the instance comprises *architecture* therefore provision the instance's behavior and two end points.

10

- To provision a behavior: given a behavior B, realize its two end points by two other end points (T1, T2) and provision (T1, T2). Find a behavior rule $B = B_1 \circ \dots \circ B_N$ and construct a sequence of logical behaviors $B_1 \circ \dots \circ B_N$ and bindings between T1 and T2. Provision each new end point in the sequence.

15

- To provision an end point: provision any end point contained by it, also if its Application Model class comprises *architecture* then realize the end points by another end point and provision that end point too.

20

The syntactical representation of the component may be as follows:

provision_instance (T1, T2, B, IM) :-

 T1 = t (Class, Instance, Id1) ,

25

 class_info (Class, Type, _) ,

 logical_link (T1, T2, -B, IM) ,

 behavior_rule (B, Bs) ,

 provision_behavior (Type, Bs, T1, T2, IM) .

30

provision_behavior (equipment, Bs, T1, T2, IM) .

-41-

provision_behavior (*architecture*, *Bs*, *T1*, *T2*, *IM*) :-

logical_link (*T1*, *T1R*, *realize*, *IM*) ,

logical_link (*T2*, *T2R*, *realize*, *IM*) ,

construct_sequence (*Bs*, *T1R*, *T2R*, *IM*) ,

5 *provision_endpoints* ([*T1R*, *T2R*], *IM*) .

construct_sequence ([*Bs*], *T1*, *T2*, *IM*) :-

logical_link (*T1*, *T2*, *-B*, *IM*) ,

construct_sequence ([*B* | *Bs*], *T1*, *T2*, *IM*) :-

10 *logical_link* (*T1*, *TA*, *-B*, *IM*) ,

logical_link (*TA*, *TB*, *bind*, *IM*) ,

provision_endpoints ([*TA*, *TB*], *IM*) ,

construct_sequence (*Bs*, *TB*, *T2*, *IM*) .

15 *provision_endpoints* ([] , *IM*) .

provision_endpoints ([*T* | *Ts*], *IM*) :-

provision_endpoints (*T*, *IM*) ,

provision_endpoints (*Ts*, *IM*) .

20 *provision_endpoints* (*T* , *IM*) :-

T = *t* (*Class*, *Instance*, *Id*) ,

class_info (*Class*, *Type*, _) ,

realize_endpoint (*Type*, *T*, *IM*)

Ts = { *TC* | *link* (*T*, *TC*, *contain*) } ,

25 *provision_endpoints* (*Ts*, *IM*) .

realize_endpoint (*equipment*, *T*, *IM*) .

realize_endpoint (*architecture*, *T*, *IM*) :-

logical_link (*T*, *T1*, *realiz* , *IM*) ,

30 *provision_endpoint* (*T1*, *IM*) .

-42-

At step 503 the inference engine's control component is used to execute the logic component. In the preferred embodiment, the algorithm utilizes artificial intelligence techniques which are intended to produce a more efficient solution to
5 a given broadband connection and planning problem. The artificial intelligence techniques include:

- 10 • Hierarchical planning: a planning problem may be decomposed into a hierarchy of sub-problems, the hierarchical planning principle solves higher level problems first, and treats those at lower levels as details to be solved later. This usually breaks down large problems into more tractable sub-problems. If a sub-problem has no solution then a way may be found to modify the higher level solutions. As a broadband connection planning problem as described according to the preferred
15 embodiment has two natural hierarchies, the Application Model class level hierarchies and instance level provisioning hierarchies, these may be used as hierarchies by an algorithm incorporating the hierarchical planning principle. If a sub-problem has no solution then higher level solutions may be modified by backtracking.
- 20 • First fail principle: during a search for a solution there is typically a conjunction of sub-goals to solve, each with a number of solutions, for example there may be several end points that must be bound to other (unspecified) end points. The first fail principle recommends selecting a
25 sub-goal with the fewest solutions. This principle results in a higher probability of detecting early failure of the search. Two special cases of the first fail principle may be particularly beneficial:

30 if any sub-goal has no solutions, then select it and backtrack immediately. The benefits of this selection is that failure of a sub-

-43-

problem should be detected as soon as possible to avoid wasted searches on other sub-problems.

if any sub-goal has exactly one solution, then select it. Selecting such a solution involves no arbitrary choices and therefore cannot lead to a combinatorial explosion, but may create new restrictions on other sub-goals which lead to earlier failure detection.

- Hill-climbing: a more intelligent variant of a depth first search that may be used if an evaluation function is available. At each point in the search an evaluation function assigns a value to alternatives, based upon an estimate of how close the search is to its goal. The alternative with the closest estimate is selected. If backtracking occurs then a next best solution is tried, and so on until all solutions have been tried. The preferred embodiment can use the hill-climbing principle with an evaluation function based upon several criteria.

The preferred embodiment's algorithm preferably places a higher priority on the first fail principle than the hierarchical principle and hill-climbing may be applied orthogonally. The preferred embodiment of the algorithm's control component preferably uses two sets of rules using its execution: computation rules and selection rules.

Fig. 6 of the accompanying drawings illustrates steps executed by the control component during step 503 of Fig. 5. At step 601 the control component uses its computation rule in order to select a sub-goal to solve. In logic programming terms a sub-goal may be an atom which is solved by matching it with a matching clause head, ie attempt to determine a set of values for variables in the sub-goal such that the truth of the sub-goal follows from the input clauses.

-44-

The computation rule decides an order in which sub-goals are solved, sub-goal order selection being dynamic, resulting in a co-routing strategy. The rules are chosen to avoid floundering, ie entering a deadlocked situation so that a situation wherein a conjunction of sub-goals of which none may be selected is
5 preferably avoided.

At step 602 the control component decides how to solve the selected sub-goals using the selection rule. At step 603 a question is asked whether more sub-goals are to be solved in order to solve the specified overall broadband
10 connection problem. This may arise due to the recursive nature of the inference engine, requiring it to select all terms in a clause as sub-goals in order to solve the problem. If the question asked at step 603 is answered in the affirmative then control passes back to step 601, otherwise control passes on to step 604 wherein the solution found may be returned as an output of the algorithm.

15

Fig. 7 of the accompanying drawings illustrates the steps executed by the computation rule during step 601 of Fig. 6. At step 701 a sub-goal may be selected according to its mode, the mode being as described hereinbelow. At step 702 a sub-goal may be selected according to its priority, the priority being as
20 described hereinbelow. At step 703 a question is asked whether selected sub-goals have equal priority. If the question asked at step 703 is answered in the negative then control is passed to step 704 wherein a sub-goal with a highest priority value is selected. If the question asked at step 703 is answered in the affirmative, the control is passed on to step 705. At step 705 the question is
25 asked whether the selected sub-goals comprise a *link* sub-goal. If the question asked at step 705 is answered in the affirmative, then any (random) sub-goal from the selected sub-goals may be selected. If the question asked at step 705 is answered in the negative then control is passed on to step 707, wherein a *link* sub-goal selection process may be executed.

30

-45-

Fig. 8 of the accompanying drawings shows the modes and priorities of sub-goals which may arise during execution of the algorithm. Modes are denoted by underlining arguments. A sub-goal may be selected if its underlined arguments are non-variables and its doubly underlined arguments are ground terms; states of its other arguments may not be considered. Priorities may be used to select among sub-goals if several may be selectable, a sub-goal with a highest priority value usually being selected first.

Priority	Subgoal and mode
4	provision_instance (<u>T1</u> , <u>T2</u> , B, IM)
4	behavior_rule (B, Bs)
4	behavior_rule_generator (B, <u>L</u>)
3	link (<u>T1</u> , T2, <u>Link</u>)
3	link (T1, <u>T2</u> , <u>Link</u>)
3	link (<u>T1</u> , <u>T2</u> , <u>Link</u>)
2	class_info (<u>Class</u> , Type, Level)
2	logical_link (T1, T2, Link, IM)
2	provision_endpoints (<u>L</u> , IM)
2	provision_endpoint (<u>T</u> , IM)
2	realize_endpoint (<u>Type</u> , <u>I</u> , IM)
2	provision_behavior (<u>Type</u> , Bs, T1, T2, IM)
2	construction_sequence (Bs, <u>T1</u> , <u>T2</u> , IM)
2	add_logiclink (<u>T1</u> , <u>T2</u> , <u>Relation</u> , IM)
2	X=Y
2	<u>X</u> = <u>Y</u>
1	integer (Id)

Fig. 9 of the accompanying drawings illustrates steps preferably executed by the algorithm during *link* sub-goal selection step 707 of Fig. 7. At step 901 a *link* sub-goal with fewest solutions is selected. This selection is an application of

-46-

the first fail principle. The selection preferably takes into consideration physical link schemata as these are usually executed in order to find all solutions, therefore it is not sufficient to merely count a number of matching clauses.

- 5 At step 902 the question is asked whether there is still a choice of *link* sub-goals. If the question asked at step 902 is answered in the affirmative then control is passed on to step 903. At step 903 the algorithm preferably selects the *link* sub-goal in an order based on a value of their *link* argument. The order of preference may be: *-Behaviour, binding, contain, realization*. This selection is an
- 10 application of the hierarchical planning principle, which postpones solving sub-problems low in the provisioning hierarchy and also those low in the containment hierarchy.

- At step 904 the question is asked whether there is still a choice of *link* sub-
- 15 goals. If the question asked at step 904 is answered in the affirmative then control is passed on to step 905. At step 905 then the *link* sub-goal with a lowest position in the class-level hierarchy, as specified by its *class_info* clause is selected. The position may be calculated by summing depths of the link's end points' instances in the class-level hierarchy (the top level having depth 0). This
- 20 selection is another application of hierarchical planning.

- At step 906 a question is asked whether there is still a choice of *link* sub-goals. If the question asked at step 906 is answered in the affirmative then control is passed on to step 907. At step 907 the *link* sub-goal may be selected
- 25 in any (random) order and control is passed onto step 908.

- If a question asked at any one of steps 902, 904 or 906 is answered in the negative then control is passed onto step 908. At step 908 the *link* sub-goal selected before control was passed to step 908 may be returned as an output of
- 30 *link* sub-goal selection step 707.

-47-

Fig. 10 of the accompanying drawings illustrates steps preferably executed by the algorithm's selection rule when it decides how to solve the selected sub-goal at step 602 of Fig. 6. The algorithm decides how to solve the selected sub-goal by choosing a clause which has a term comprising the sub-goal. At step 1001 a question is asked whether a selected sub-goal is a *link* sub-goal. If the question asked at step 1001 is answered in the negative then control is passed on to step 1002. At step 1002 a clause which the algorithm select in order to attempt to solve the sub-goal is chosen according to its textual order, ie its position in the algorithm's input clause list. The result of step 1002 may then be returned as an output of step 602.

If the question asked at step 1001 is answered in the affirmative then control is passed to step 1003. At step 1003 an unbound argument of the *link* sub-goal is selected. The unbound argument may be called a "target". A clause whose target already has a logical link is preferably selected.

At step 1004 a question is asked whether there is still a choice of clauses. If the question asked at step 1004 is answered in the affirmative then control is passed on to step 1005. At step 1005 a clause whose target is lowest in the class-level hierarchy is selected. This selection is an application of the hill-climbing principle which aims to achieve provisioning with as few instances as possible.

At step 1006 a question is asked whether there is still a choice of clauses. If the question asked at step 1006 is answered in the affirmative then control is passed on to step 1007. At step 1007 a clause whose target capacity is smallest is selected. This selection is another application of the hill-climbing principle which aims to minimize wasted end point capacity.

30

-48-

At step 1008 a question is asked whether there is still a choice of clauses. If the question asked at step 1008 is answered in the affirmative then control is passed on to step 1009. At step 1009 a clause is selected in any (random) order and control is passed on to step 1010.

5

If any one of the questions asked at step 1004, 1006 or 1008 is answered in the negative, control is passed on to step 1010. At step 1010 the clause selected by the algorithm before control was passed on to step 1010 may be returned as an output of step 602.

10

There follows a condensed trace showing how the example of Fig. 4 entered into the algorithm in the syntactical representation described hereinabove may be solved. For simplicity, the trace does not demonstrate the algorithm's selection rule, nor all the artificial intelligence techniques used to guide the inference engine's search, but it does illustrate the co-routing nature of the computation rule, interleaving of logical link selection and behavior rule construction. However, it will be appreciated by those skilled in the art that the algorithmic steps illustrated with reference to Fig. 7, Fig. 9 and Fig. 10 hereinabove could be implemented and demonstrated using a substantially similar trace technique. For readability, some intermediate steps have been omitted from the trace, again, those skilled in the art will appreciate that these details may be inferred from material disclosed. Herein variable *IM* which contains the algorithm's representation of the Implementation Model is shown as an unbound variable (for brevity) but its contents shall be summarized whenever they change (whenever an *add_logical_link* sub-goal is solved).

The trace preferably starts with initial goal (to provision *tdm* instance 401):

provision_instance (*t* (*tdm*, 1, 1), *t* (*tdm*, 1, 2), *adapt*, *IM*)

30

-49-

and no logical links in *IM* (*IM* = []). Select *provision_instance* (more precisely, resolve atom with that predicate symbol):

```

class_info (tdm, Type, _) ,
5  T1=t (tdm, 1, Id1) ,
   T2=t (tdm, 1, Id2) ,
   logical_link (T1, T2, -B, IM) ,
   behavior_rule (B, Bs) ,
   provision_behavior (type, Bs, T1, T2, IM) ,
10  provision_endpoints ( [T1, T2], IM) ,

```

Select *class_info*, = (twice), *logical_link*, *behavior_rule*, *provision_endpoints*:

```

link (t (tdm, 1, Id1) , t (tdm, 1, Id2) , -B) ,
15  add_logical_link (t (tdm, 1, Id1) , t (tdm, 1, Id2) , -B, IM) ,
   behavior_rule_generator (B, BS1) ,
   provision_behavior (architecture, [B|Bs1] , t (tdm, 1, Id1) , t (tdm, 1, ID2), IM)
   provision_endpoint (t (tdm, 1, ID1), IM) ,
   provision_endpoint (t (tdm, 1, ID2), IM) ,
20

```

Now there is a *behavior_rule_generator* sub-goal which cannot yet be solved because its list is unbound. The algorithm is guided by logical links but each logical link is immediately tested for consistency against the partially-constructed behavior rule, which is currently:

```

25  B = B o Bs1

```

where *B* and *Bs1* are unbound.

```

30  Now select link and provision_behavior.

```

-50-

add_logical_link (*t* (*tdm*, 1, 1) , *t* (*tdm*, 1, 2), -*adapt*, *IM*) ,
behavior_rule_generator (*adapt*, *Bs1*) ,
logical_link (*t* (*tdm*, 1, 1), *T1R*, *realize*, *IM*) ,
5 *logical_link* (*t* (*tdm*, 1, 2), *T2R*, *realize*, *IM*) ,
construct_sequence ([*adapt* | *Bs1*], *T1R*, *T2R*, *IM*) ,
provision_endpoints ([*T1R*, *T2R*, *IM*] ,
provision_endpoints (*t* (*tdm*, 1, 1), *IM*) ,
provision_endpoints (*t* (*tdm*, 1, 2), *IM*) .

10

Solving *link* has bound *B* to *adapt* so the behavior rule is now:

adapt = *adapt* o *Bs1*

15

Now select *add_logical_link*, *logical_link* (twice) and *provision_endpoints*:

behavior_rule_generator (*adapt*, *Bs1*) ,
link (*t* (*tdm*, 1, 1) , *T1R*, *realize*, *IM*) ,
add_logical_link (*t* (*tdm*, 1, 1) , *T1R*, *realize*, *IM*) ,
20 *link* (*t* (*tdm*, 1, 2) , *T2R*, *realize*, *IM*) ,
add_logical_link (*t* (*tdm*, 1, 2) , *T2R*, *realize*, *IM*) ,
construct_sequence ([*adapt* | *Bs1*] , *T1R*, *T2R*, *IM*) ,
provision_endpoint (*T1R*, *IM*) ,
provision_endpoint (*T2R*, *IM*) ,
25 *provision_endpoint* (*t* (*tdm*, 1, 1), *IM*) ,
provision_endpoint (*t* (*tdm*, 1, 2), *IM*) .

Since *add_logical_link* was solved there is now a nonempty *IM*:

30

t (*tdm*, 1, 1) *adapt* *t* (*tdm*, 1, 2)

-51-

which is the logical *adapt* behavior 404 between the endpoints 1 and 2 of TDM instance 401. Now select both *link* sub-goals:

```

5      behavior_rule_generator (adapt, Bs1) ,
      add_logical_link (t (tdm, 1, 1), t (cell_ace, 1, 1), realize, IM),
      construct_sequence ( [adapt | Bs1], t (cell_ace, 1, 1) , t (avj, 1, 2), IM) ,
      provision_endpoint (t (cell_ace, 1, 1), IM) ,
      provision_endpoint (t (avj, 1, 2), IM) ,
10     provision_endpoint (t (tdm, 1, 1), IM) ,
      provision_endpoint (t (tdm, 1, 2), IM) ,
      provision_endpoint (t (avj, 1, 2), IM) ,
      provision_endpoint (t (tdm, 1, 1), IM) ,
      provision_endpoint (t (tdm, 1, 2), IM)

```

15

The *IM* is now:

```

      t (tdm, 1, 1) adapt t (tdm, 1, 2)
      t (tdm, 1, 1) realize t (cell_ace, 1, 1)
20     t (tdm, 1, 2) realize t (avj, 1, 2)
      t (cell_ace, 1, 1) adapt t (cell_ace, 1, 2)

```

Select *add_logical_link* and *construct_sequence*:

```

25     behavior_rule_generator (adapt, [B] ) ,
      provision_endpoint (t (cell_ace, 1, 2) , IM) ,
      provision_endpoint (t (avj, 1, 1) , IM) ,
      logical_link (t (avj, 1, 1) , t (avj, 1, 2), -B, IM) ,
      provision_endpoint (t (cell_ace, 1, 1) , IM) ,
30     provision_endpoint (t (avj, 1, 2) , IM) ,

```

-52-

provision_endpoint (*t* (*tdm*, 1, 1) , *IM*) ,
provision_endpoint (*t* (*tdm*, 1, 2) , *IM*)

The current behavior rule is now:

5

adapt = *adapt* o *B*

and the *IM* is now:

10

t (*tdm*, 1, 1) *adapt* *t* (*tdm*, 1, 2)
t (*tdm*, 1, 1) *realize* *t* (*cell_ace*, 1, 1)
t (*tdm*, 1, 2) *realize* *t* (*avj*, 1, 2)
t (*cell_ace*, 1, 1) *adapt* *t* (*cell_ace*, 1, 2)
t (*cell_ace*, 1, 2) *bind* *t* (*avj*, 1, 1)

15

Select *logical_link* and *behavior_rule_generator*:

20

provision_endpoint (*t* (*cell_ace*, 1, 2) , *IM*) ,
provision_endpoint (*t* (*avj*, 1, 1) , *IM*) ,
link (*t* (*avj*, 1, 1), *t* (*avj*, 1, 2, -*adapt*) ,
add_logical_link (*t* (*avj*, 1, 2), -*adapt*, *IM*) ,
provision_endpoint (*t* (*cell_ace*, 1, 1) , *IM*) ,
provision_endpoint (*t* (*avj*, 1, 2) , *IM*) ,
provision_endpoint (*t* (*tdm*, 1, 1) , *IM*) ,
provision_endpoint (*t* (*tdm*, 1, 2) , *IM*) .

25

The behavior rule is now completely constructed:

adapt = *adapt* o *adapt*

30

-53-

Select *add_logical_link* and *link*:

5 *provision_endpoint* (*t* (*cell_ace*, 1, 2) , *IM*) ,
 provision_endpoint (*t* (*avj*, 1, 1) , *IM*) ,
 provision_endpoint (*t* (*cell_ace*, 1, 1) , *IM*) ,
 provision_endpoint (*t* (*avj*, 1, 2) , *IM*) ,
 provision_endpoint (*t* (*tdm*, 1, 1) , *IM*) ,
 provision_endpoint (*t* (*tdm*, 1, 2) , *IM*).

10 The *IM* is now:

t (*tdm*, 1, 1) *adapt* *t* (*tdm*, 1, 2)
 t (*tdm*, 1, 1) *realize* *t* (*cell_ace*, 1, 1)
 t (*tdm*, 1, 2) *realize* *t* (*avj*, 1, 2)
 15 *t* (*cell_ace*, 1, 1) *adapt* *t* (*cell_ace*, 1, 2)
 t (*cell_ace*, 1, 2) *bind* *t* (*avj*, 1, 1)
 t (*avj*, 1, 1) *adapt* *t* (*avj*, 1, 2)

 since the class type of each of the *cell_ace* and *avj* instances is equipment,
 20 all the sub-goals may be solved except:

provision_endpoint (*t* (*tdm*, 1, 1))
 provision_endpoint (*t* (*tdm*, 1, 2))

25 Solving these sub-goals leads to the contained endpoints *t* (*tdm*, 1, 3) and *t* (*tdm*, 1, 4). The logical behavior *adapt* is found between them and they are provisioned exactly as above, except that no further endpoint and the TDM has been provisioned with the following Implementation Model, which for the example of Fig. 4 comprises all physical links selected as logical links:

30

-54-

IM	link's label in Fig. 4
<i>t (tdm, 1, 1) adapt t (tdm, 1, 2)</i>	404
<i>t (tdm, 1, 1) realize t (cell_ace, 1, 1)</i>	418
<i>t (tdm, 1, 2) realize t (avj, 1, 2)</i>	419
<i>t (cell_ace, 1, 1) adapt t (cell_ace, 1, 2)</i>	420
<i>t (cell_ace, 1, 2) bind t (avj, 1, 1)</i>	421
<i>t (avj, 1, 1) adapt t (avj, 1, 2)</i>	422
<i>t (tdm, 1, 1) contain t (tdm, 1, 3)</i>	423
<i>t (tdm, 1, 2) contain t (tdm, 1, 4)</i>	424
<i>t (tdm, 1, 3) adapt t (tdm, 1, 4)</i>	407
<i>t (tdm, 1, 3) realize t (cell_ace, 1, 3)</i>	426
<i>t (tdm, 1, 4) realize t (avj, 1, 4)</i>	427
<i>t (cell_ace, 1, 3) adapt t (cell_ace, 1, 4)</i>	428
<i>t (cell_ace, 1, 4) bind (avj, 1, 3)</i>	429
<i>t (avj, 1, 3) adapt t (avj, 1, 4)</i>	430

There now follows a second example of the execution of the algorithm solving another broadband connection problem. Fig. 11 of the accompanying drawings illustrates a graph which represents a simplified Application Model. All physical bindings and realizations for the Application Model are shown on the graph, although physical behaviors are not shown. To simplify the description of the diagram and the execution of the algorithm on a syntactical representation of the nodes and links shown in the graph, the following points should be noted:

- behavior rules are ignored
- there is no end point containment, we assume that contained end points (for example those of type SS7) do not exist. End points are shown as numerals near class instances (shown as oval boxes). Again, realization

-55-

links are illustrated as dotted lines with an arrow head and binding links are shown as solid lines.

The example of Fig. 11 comprises 6 classes: *x*, *atm_switch*, *tdm*,
5 *frame_processor*, *cell_ace* and *avj*. Class *x* is at a top level of the class-level hierarchy; *tdm*, *frame_processor* and *atm_switch* are on a level under *x* and *avj* and *cell_ace* are on a third level below. A first instance of the *x* class 1101 may have its end point 1 realized (1115) by end point 1 of a first *tdm* instance 1102 and also realized (1113) by end point 1 of a second *tdm* instance 1103. End
10 point 2 of *x* instance 1101 may be realized (1114) by end point 2 of *frame_processor* instance 1105.

End point 1 of *tdm* instance 1102 may be realized (1116) by end point 1 of a first *cell_ace* instance 1106. End point 2 of *tdm* instance 1102 may be realized
15 (1117) by end point 2 of a first *avj* instance 1107. End point 1 of *tdm* instance 1103 may be realized (1119) by end point 1 of a second *cell_ace* instance 1108. End point 2 of *tdm* instance 1103 may be realized (1118) by end point 2 of a second *avj* instance 1109. End point 2 of *cell_ace* instance 1108 and end point 1 of *avj* instance 1109 are linked by binding 1128.

20

End point 1 of *frame_processor* instance 1105 is linked by binding 1120 to end point 2 of *atm_switch* instance 1104. End point 1 of *atm_switch* instance is linked to end point 2 of a third *avj* instance 1110 by binding 1121. End point 3 of *atm* switch instance 1104 is linked to end point 2 of a fourth *avj* instance 1111 by
25 binding 1122. End point 4 of *atm_switch* instance 1104 is linked to end point 2 of a fifth *avj* instance 1112 by binding 1123.

End point 1 of *avj* instance 1110 is linked to end point 2 of *tdm* instance 1102 by binding 1125. End point 1 of *avj* instance 1111 is linked to end point 2 of

-56-

tdm instance 1102 by binding 1126. End point 1 of *avj* instance 1112 is linked to end point 2 of *tdm* instance 1103 by binding 1127.

The syntactical representation of the application model shown in Fig. 11 may be substantially similar to the syntactical representation method described hereinabove. For example, clauses defining the example's classes may be as follows:

```

10  class_info (x, architecture, 0)
    class_info (tdm, equipment, 1)
    class_info (frame_processor, equipment, 1)
    class_info (atm_switch, equipment, 1)
    class_info (avj, equipment, 2)
    class_info (cell_ace, equipment, 2)

```

15

It will be appreciated by those skilled in the art that end point definitions according to the preferred embodiment may also be created for nodes of Fig. 11, for example, end point 2 of the third *avj* instance 1110 may be represented by the following clause:

20

```

    endpoint (avj, 3, 2)

```

Links between the nodes may also be created using the syntactical representation, for example, the realize link 1116 between end point 1 of the first *tdm* instance 1102 and end point 1 of the first *cell_ace* instance 1106 may be represented by the following clause:

25

```

    link (t (tdm, 1, 1) , t (cell_ace, 1, 1) , realize) .

```

-57-

Figs. 12 to 15 of the accompanying drawings along with the corresponding description given hereinbelow will demonstrate the execution of the algorithm according to the preferred embodiment, including use of the artificial intelligence techniques such as first fail, hill-climbing and hierarchical planning. The result of the algorithm's execution will again be an Implementation Model (IM) comprising selected logical links, which may be expressed according to the syntactical representation.

Fig. 12 of the accompanying drawings illustrates steps which may be performed by the preferred embodiment of the algorithm whilst attempting to solve a broadband connection problem on the application model of Fig. 11 up to a first occurrence of the algorithm back-tracking. At step 1201 a provisioning instance to be solved is specified. In the following example, *x* instance 1101 will be provisioned. The syntactical representation of the specific instance to be provisioned may be given by the following clause:

provision_instance (t (x, 1, 1) , t (x, 1, 2) , adapt, IM)

x instance 1101 has two end points (numbered 1 and 2) which have a behavior link (not shown in Fig. 11). Both the end points should be realized in order to solve the broadband connections planning problem. End point 1 of *x* instance 1201 may be realized in two ways (ie by *tdm* instances 1106 and 1103), whereas end point 2 has only one possible realization (*frame_processor* instance 1105). The algorithm's application of the first fail principle causes end point 2 to be selected and realized by realization link 1114 to end point 2 of *frame_processor* 1105. This may be expressed in the syntactical representation by the following clause:

link (t (x, 1, 1) , t (frame_processor, 1, 2), realize)

30

-58-

At step 1203 the algorithm's application of the first fail principle selects the behavior linking *frame_processor* instance's 1105 end points 1 and 2. This may be expressed as a logical link to be added to the algorithm's Implementation Model in the syntactical representation by the following clause:

5

link (t (frame_processor, 1, 2) , t (frame_processor, 1, 1), behavior)

At step 1204 the algorithm's application of the first fail principle binds end point 1 of *frame_processor* instance 1105 to end point 2 of *atm_switch* instance 1104 using binding link 1120.

10

link (t (frame_processor, 1, 1) , t (atm_switch, 1, 2), bind)

As *atm_switch* instance 1105 end point 2 has three possible behavior links (linking it with end points 1, 3 and 4 of *atm_switch* instance 1105). However, end point 1 of *x* instance 1101 has only two possible realizations (linking it to end point 1 of *tdm* instance 1102 or end point 1 of *tdm* instance 1103). The algorithm's application of the first fail principle therefore decides that end point 1 of the *x* instance is to be attempted to be provisioned, rather than end point 2 of *atm_switch* instance 1104. As the algorithm has no reason to select end point 1 of *tdm* instance 1102 over end point 1 of *tdm* instance 1103 or vice versa, the algorithm (randomly) selects end point 1 of *tdm* instance 1102, realized by realization link 1115.

20

link (t (x, 1, 1) , t (tdm, 1, 1), realize)

25

At step 1206 the algorithm's application of the hill-climbing principle tries behaviors before bindings, so end points 1 and 2 of the *tdm* instance 1102 are linked by their behavior link:

30

-59-

link (t (tdm, 1, 1) , t (tdm, 1, 2), behavior)

At step 1207 end point 2 of *tdm* instance 1102 has two possible bindings (to end point 1 of *avj* instance 1110 and end point 1 of *avj* instance 1111), so the algorithm's application of the first fail principle realizes (1116) end point 1 of *tdm* instance 1102 by end point 1 of *cell_ace* instance 1106:

link (t (tdm, 1, 1) , t (class_instance, 1, 1) , realize)

10 At step 1208 the algorithm's application of the first fail principle selects a behavior link linking end points 1 and 2 of *cell_ace* instance 1106:

link (t (cell_ace, 1, 1) , t (cell_ace, 1, 2) , behavior)

15 At this point, there are no possible bindings on end point 2 of *cell_ace* instance 1106, so the algorithm's application of the first fail principle causes immediate back-tracking. Fig. 13 of the accompanying drawings illustrates schematically steps 1201 to 1208 of Fig. 12 performed on the simplified Application Model of Fig. 11. Logical links selected are shown as emboldened physical links, apart from logical behaviors (between end points of a single instance) which are implied by emboldened instance boxes. Numerals 1201 to 20 1208 on Fig. 13 show the order of selection corresponding to the steps of Fig. 12.

Fig. 14 of the accompanying drawings illustrates the steps preferably 25 executed by the algorithm following step 1208 of Fig. 12. At step 1401 the algorithm back tracks to a choice point occurring at step 1205. At step 1402 the algorithm tries an alternative realization of end point 1 of *x* instance 1101 by realizing (1113) the end point by end point 1 of *tdm* instance 1103:

30 *link (t (x, 1, 1) , t (tdm, 2, 1) , realize)*

-60-

At step 1403 the algorithm's application of the first fail principle links end points 1 and 2 of *tdm* instance 1103 by a behavior:

5 *link (t (tdm, 2, 1) , t (tdm, 2, 2) , -behavior)*

At step 1404 the algorithm's application of the hierarchical planning principle binds end point 2 of *tdm* instance 1103 to end point 1 of *avj* instance 1112 by binding link 1127:

10

link (t (tdm, 2, 2) , t (avj, 5, 1) , bind)

At step 1405 the algorithm's application of the hierarchical planning principle links end points 1 and 2 of *avj* instance 1112 by a behavior.

15

link (t (avj, 5, 1) , t (avj, 5, 2) , behavior)

At step 1406 the algorithm's application of the hierarchical planning principle binds end point 2 of *avj* instance 1112 to end point 4 of *atm_switch* instance 1104 by binding link 1123.

20

link (t (avj, 5, 2) , t (atm_switch, 1, 4) , bind)

At step 1407 the algorithm's application of the hierarchical planning principle delays provisioning of *tdm* instance 1103 in order to construct links to *avj* instance 1112 and *atm_switch* instance 1114. However, the algorithm's application of the first fail principle overrides its use of the hierarchical planning principle in this case and delays selection of a behavior linking *atm_switch* instance 1104 and instead provisions *tdm* instance 1103. End point 1 of *tdm* instance 1103 is realized (1119) by end point 1 of *cell_ace* instance 1108:

30

-61-

link (t (avj, 5, 2) , t (atm_switch, 1, 4) , bind)

At step 1408 the algorithm's application of the first fail and hierarchical
5 planning principles link end points 1 and 2 of *cell_ace* instance 1108 by a
behavior:

link (t (cell_ace, 2, 1) , t (cell_ace, 2, 2) , behavior)

10 At step 1410 the algorithm's application of the first fail and hierarchical
planning principles bind end point 2 of *cell_ace* instance 1108 to end point 1 of
avj instance 1109 by binding link 1128:

link (t (cell_ace, 2, 2) , t (avj, 2, 1) , bind)

15

At step 1411 the algorithm's application of the first fail and hierarchical
planning principles link end points 1 and 2 of *avj* instance 1109 by a behavior.

link (t (avj, 2, 1) , t (avj, 2, 2) , behavior)

20

At step 1411 the algorithm's application of the first fail principle realizes
(1118) end point 2 of *tdm* instance 1103 by end point 2 of *avj* instance 1109.

link (t (avj, 2, 2) , t (tdm, 2, 2) , realize)

25

At step 1412 the only two end points which still require logical links are end
points 2 and 4 of *atm_switch* instance 1104:

link (t (atm_switch, 1, 2) , t (atm_switch, 1, 4) , -behavior)

30

-62-

The algorithm's application of the hill-climbing principle selects a behavior between the two end points because the target already had a logical link. Without this heuristic different behaviors might have been invoked (eg linking end points 4 and 3 or end points 2 and 1 of the *atm_switch* instance 1104) causing the two ends of the specified instance to be provisioned to "miss" each other and leading to unnecessary back-tracking.

Fig. 15 of the accompanying drawings illustrates a trace of the execution of the algorithm in a manner similar to Fig. 13, including steps 1201 to 1205 of Fig. 12 and steps 1401 to 1412 of Fig. 14.

Thus, the Implementation Model resulting from the execution of the algorithm as described hereinabove may be as follows:

IM	link's label in Fig. 11
<i>t</i> (x, 1, 1) realize <i>t</i> (frame_processor, 1, 2)	1114
<i>t</i> (frame_processor, 1, 2) behavior <i>t</i> (frame_processor, 1, 1)	1105
<i>t</i> (frame_processor, 1, 1) bind <i>t</i> (atm_switch, 1, 2)	1120
<i>t</i> (x, 1, 1) realize <i>t</i> (tdm, 2, 1)	1113
<i>t</i> (tdm, 2, 1) behavior <i>t</i> (tdm, 2, 2)	1103
<i>t</i> (tdm, 2, 2) bind <i>t</i> (avj, 5, 1)	1127
<i>t</i> (avj, 5, 1) behavior <i>t</i> (avj, 5, 2)	1112
<i>t</i> (avj, 5, 2) bind <i>t</i> (atm_switch, 1, 4)	1123
<i>t</i> (avj, 5, 2) realize <i>t</i> (cell_ace, 2, 1)	1119
<i>t</i> (cell_ace, 2, 1) behavior <i>t</i> (cell_ace, 2, 2)	1108
<i>t</i> (cell_ace, 2, 2) bind <i>t</i> (avj, 2, 1)	1128
<i>t</i> (avj, 2, 1) behavior <i>t</i> (avj, 2, 2)	1109
<i>t</i> (avj, 2, 2) realize <i>t</i> (tdm, 2, 2)	1118
<i>t</i> (atm_switch, 1, 2) behavior <i>t</i> (atm_switch, 1, 4)	1104

15

-63-

The logical links in the resulting implementation model could be displayed to a user, preferably via the network controller in any convenient form, for example a screen display highlighting the logical links on a graphical representation of the graph similar to that shown in fig. 15 or alternatively a list of clauses in the syntactical representation describing the Implementation Model's solution. The
5 resulting Implementation Model could also be used, for example, by a network controller to implement the logical links for connections in the physical network.

-64-

Claims:

1. In a network comprising a plurality of nodes and links, each said node having at least one end point connecting it to a said link element, each said link between said end points having a type, a method of producing a connection
5 plan representing at least one connection between said end points in said network, said connection plan comprising data describing a set of said end points and said type of said link between the end points said method comprising the steps of:

10 creating a data representation of at least part of said network;

creating a data representation of end points of said connection; and

15 applying an algorithm to said data representations to generate said connection plan.

2. A method according to claim 1, wherein a said representation comprises a list of at least one clause, each said clause comprising at least one term.

20

3. A method according to claim 2, wherein said clauses comprise assertions and implications.

4. A method according to claim 2, wherein said algorithm selects a
25 said term of a said clause representing said connection as a sub-goal and recursively attempts to solve said sub-goal.

5. A method according to claim 4, wherein a said sub-goal is assigned a priority, and said algorithm selects a said sub-goal with a highest said priority.

30

-65-

6. A method according to claim 4, wherein said algorithm selects a said sub-goal which is a clause with fewest said terms.

7. A method according to claim 4, wherein said algorithm selects a
5 said sub-goal according to order of its said terms.

8. A method according to claim 4, wherein a said algorithm selects a sub-goal with a lowest position in said network hierarchy.

10 9. A method according to claim 4, wherein said algorithm selects said sub-goal randomly.

10. A method according to claim 4, wherein said algorithm selects a said clause with which to solve said sub-goal according to position of said clause
15 in said list of clauses.

11. A method according to claim 4, wherein said algorithm selects a said clause with which to solve said sub-goal by selecting a said clause having a term included in said connection plan.

20

12. A method according to claim 4, wherein said algorithm selects a said clause with which to solve said sub-goal by selecting a clause which is lowest in said network hierarchy.

25 13. A method according to claim 4, wherein said algorithm selects a said clause with which to solve said sub-goal by selecting a clause with a term having smallest capacity.

14. A method according to claim 4, wherein said algorithm randomly
30 selects a clause with which to solve said sub-goal.

-66-

15. A method according to claim 2, wherein said clauses represent data describing features of said network, said features selected from the set:

5 classes of said network nodes;

end points of said network nodes;

links of said network;

10

schematic representations of said link types.

16. A method according to claim 1, wherein said link types may be selected from the set:

15

binding;

realization;

20

behavior;

containment.

17. A method according to claim 1, further comprising the step of:

25

displaying a graphical representation of said connection plan.

18. A method according to claim 1, further comprising the step of:

30

implementing said connection plan as a connection in said network.

-67-

19. A method according to claim 1, wherein said representation comprises a substantially Prolog-like syntax.

5 20. In a network comprising a plurality of nodes and links, each said node having at least one end point connecting it to a said link element, each said link between said end points having a type, connection planning apparatus for planning at least one connection between said end points in said network by creating a connection plan comprising a set of said end points and said types of
10 link between the end points said connection planning apparatus comprising:

means of creating a representation of at least part of said network;

means of creating a representation of end points of said connection; and

15

an inference engine which uses said representation to generate said connection plan.

21. Apparatus according to claim 20, wherein said representation
20 comprises a list of at least one clause, each said clause comprising at least one term.

22. Apparatus according to claim 21, wherein said clauses comprise
horn clauses.

25

23. Apparatus according to claim 21, wherein said inference engine selects a said term of a said clause representing said connection as a sub-goal and recursively attempts to solve said sub-goal.

-68-

24. Apparatus according to claim 23, wherein a said sub-goal is assigned a priority, and said inference engine selects a said sub-goal with a highest said priority.

5 25. Apparatus according to claim 23, wherein said inference engine selects a said sub-goal which is a clause with said fewest said terms.

26. Apparatus according to claim 23, wherein said inference engine selects a sub-goal according to order of its said terms.

10

27. Apparatus according to claim 23, wherein said inference engine selects said sub-goal with a lowest position in said network hierarchy.

15 28. Apparatus according to claim 23, wherein said inference engine selects said sub-goal randomly.

29. Apparatus according to claim 23, wherein said inference engine selects a said clause with which to solve said sub-goal according to position of said clause in said list of clauses.

20

30. Apparatus according to claim 23, wherein said inference engine selects a said clause with which to solve said sub-goal by selecting said clause having a term included in said connection plan.

25 31. Apparatus according to claim 23, wherein said inference engine selects a said clause with which to solve said sub-goal by selecting a clause which is lowest in said network hierarchy.

-69-

32. Apparatus according to claim 23, wherein said inference engine selects a said clause with which to solve sub-goal by selecting a clause with a term having smallest capacity.

5 33. Apparatus according to claim 23, wherein said inference engine randomly selects a clause with which to solve said sub-goal.

34. Apparatus according to claim 21, wherein said clauses represent data describing features of said network, said features selected from the set:

10

classes of said network nodes;

end points of said network nodes;

15

links of said network;

schemata representing link types.

35. Apparatus according to claim 20, wherein said link types may be
20 selected from the set:

binding;

realization;

25

behavior;

containment.

30

36. Apparatus according to claim 20, further comprising:

-70-

means for displaying a graphical representation of said connection plan.

5 37. Apparatus according to claim 20, further comprising:

means for implementing said connection plan as a connection in said network.

10 38. Apparatus according to claim 20, wherein said representation comprises a substantially Prolog-like syntax.

15 39. A syntactical representation for describing capabilities and links in a hierarchical network, said syntactical representation comprising a list of at least one clause.

40. A syntactical representation according to claim 39, wherein said capability representation comprises:

20 a node identifier;
an input data type;

an output data type; and

25 a hierarchical network layer upon which said node appears.

41. A syntactical representation according to claim 39, wherein said syntactical representation comprises:

30

-71-

at least two nodes representing end points of said links; and

a hierarchical network layer upon which said link appears.

5 42. A syntactical representation according to claim 39, comprising a substantially Prolog-like syntax.

43. An inference engine for solving connection problems in a hierarchical network comprising:

10

a logic component comprising a specification of what it means to solve said problem; and

a control component comprising a description of how said logic component
15 is to be executed.

44. An inference engine according to claim 43, wherein said control component comprises:

20 a computation rule for selecting sub-goals with which to solve said problems; and

a selection rule for selecting clauses with which to solve said selected sub-goals.

25

45. An inference engine according to claim 43, wherein said control component executes said logic component by applying techniques selected from the set:

30 hierarchical planning;

-72-

hill-climbing;

first fail.

5

46. A network management apparatus for a communications network comprising a plurality of interconnected network element devices, said network management apparatus comprising:

10 a plurality of device controllers each controlling at least one corresponding network element device, said plurality of device controllers comprising a plurality of device sub-controllers,

wherein a plurality of functions performed by a network element device are
15 each represented by a corresponding respective said device sub-controller; and

said plurality of device controllers are arranged in a manner which represents an arrangement of a service provided by said network element devices.

20

47. The network management apparatus as claimed in claim 46, wherein each said device sub-controller comprises:

a data storage means storing data describing a function of a network
25 element device; and a means for generating management control signals for controlling a said network element device.

48. The device sub-controller as claimed in claim 46, wherein a said device sub-controller comprises an object data representation.

30

-73-

49. The network management apparatus as claimed in any one of claims 46 to 48, wherein a set of connections between said device controllers are provided, said connections arranged in a manner which represents service connections between said network element devices.

5

49. In a network management apparatus, a method of controlling a network comprising a plurality of network element devices each supporting a plurality of services implemented by functions of the network element devices, said method comprising the steps of:

10

representing each said function supported by a network element device by a corresponding respective device sub-controller by;

at each said sub-controller storing data signals representing a said function
15 capability of said network element device; and

20

operating said sub-controllers to generate a plurality of management control signals for controlling said network element device to provide said functionality in supporting said services.

50. In a network management apparatus, a method of controlling a network comprising a plurality of network element devices each supporting a plurality of service functions, said method comprising the steps of:

25 representing each network element device by a corresponding device controller;

for each device controller, representing each function capability provided by a said corresponding network element device by a corresponding device sub-
30 controller; and

-74-

linking a plurality of said sub-controllers of different device controllers in a manner which represents a service connection supported by said network element devices.

5

51. A method of constructing a management apparatus for a communications network comprising a plurality of node element devices, said method comprising the steps of:

10 representing a plurality of network element devices of the network by a corresponding plurality of function models, wherein each function model comprises a plurality of specifications of functionality of a said corresponding network element device; and

15 implementing said function models as a plurality of device controllers each controlling a corresponding respective said network element device, wherein each said device controller comprises a plurality of device sub-controllers each device sub-controller corresponding to a service function provided by a network element device controlled by said device controller.

20

52. The method as claimed in claim 51, wherein each said device sub-controller comprises:

a data storage medium for storing a set of data signals describing a
25 functionality of a network element device; and

a control signal generating means for generating control signals for controlling a function of said network element device.

-75-

53. In a communications network comprising a plurality of interconnected nodes, each node providing a plurality of services, a method of planning a service between a source node and at least one destination node comprising the steps of:

5

storing data representing connections between said nodes;

storing data representing services provided by said nodes;

10 finding data representing a set of connections and services between said source and destination nodes which implement said service; and

transforming said data representing a set of connections and services for application within a network controller.

15

54. A method as claimed in claim 53, wherein a said service provided by a node comprises transforming a first service type to a second service type.

55. A method as claimed in claim 53, wherein said data representing
20 connections comprises a name of a first node, a name of a second node and a name of a layer within which said nodes are contained.

56. A method as claimed in claim 53, wherein said data representing
services comprises a name of a node, a name of a first service type provided by
25 said node and a name of a second service type provided by said node.

57. A method as claimed in claim 53, wherein said data transformed for application within a network controller comprises an aggregated connection between said network nodes for implementing a said service.

30

-76-

58. Apparatus comprising a processor and a data storage means configured to execute method as claimed in any of the above claims 53 to 57.

59. In a communications network comprising a plurality of inter-
5 connected nodes, said nodes supporting at least one service function, a method of planning a connection supporting said service, said method comprising the steps of:

translating function capabilities of said nodes and connections between said
10 nodes into a plurality of syntax constructs;

taking each said syntax construct as an operator in a planning system to obtain a connection plan solution; and

15 transforming said connection plan solution into a connection model suitable for application within a network manager apparatus.

60. The method as claimed in claim 59, wherein a said syntactical operator comprises a connection operator.
20

61. The method as claimed in claim 59 or 60, wherein a said syntactical operator comprises a transform operator representing a said function capability of a said node.

25 62. The method as claimed in any one of claims 59 to 61, wherein a said service is represented in terms of a layered structure.

63. The method as claimed in any one of claims 59 to 62, wherein said connection plan solution is obtained using an artificial intelligence technique.
30

-77-

64. The method as claimed in any one of claims 59 to 63, wherein a said connection comprises a multi-media service connection.

1/10

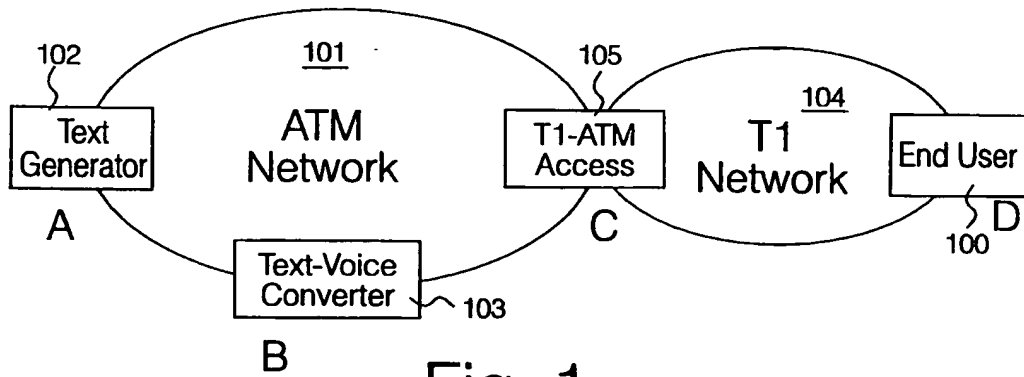


Fig. 1

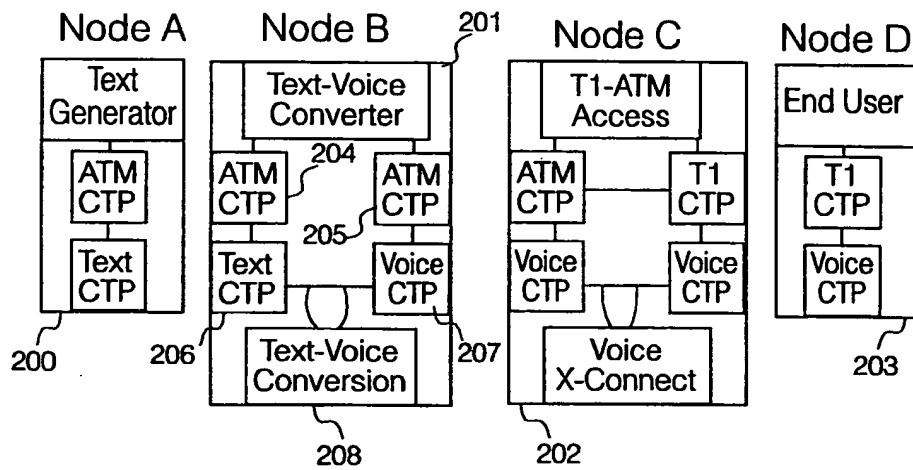


Fig. 2

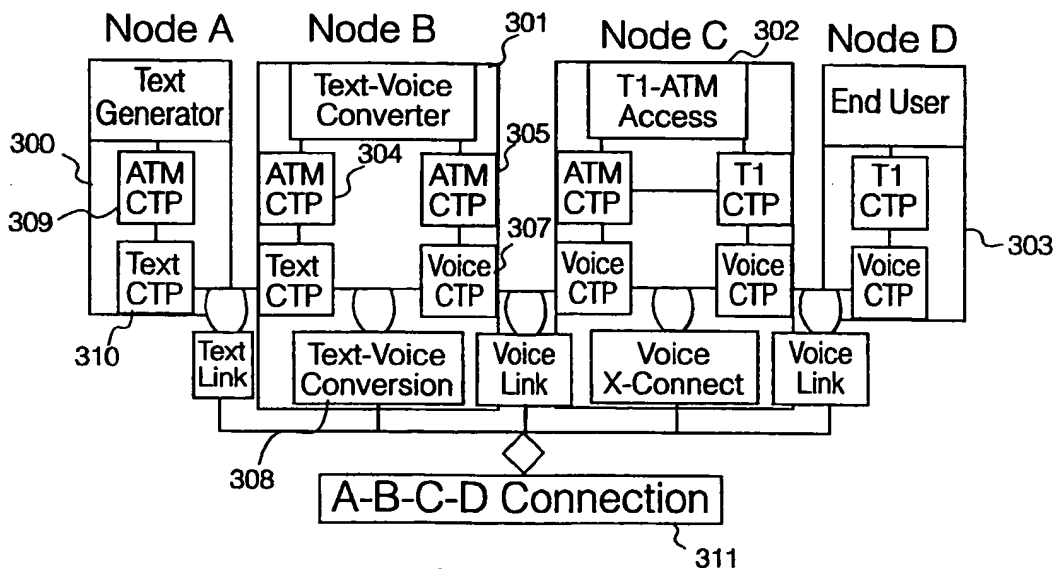


Fig. 3

2/10

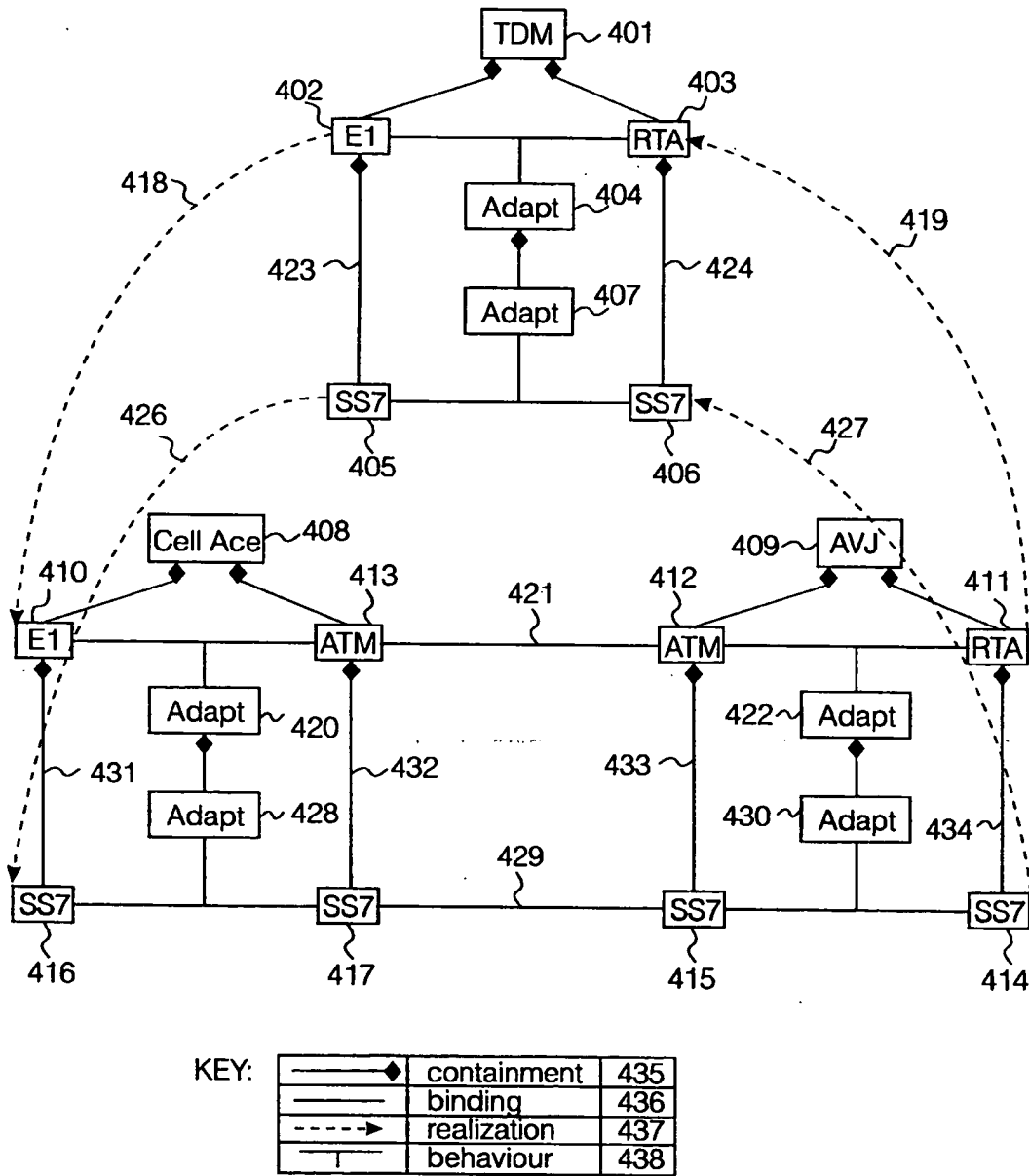


Fig. 4

3/10

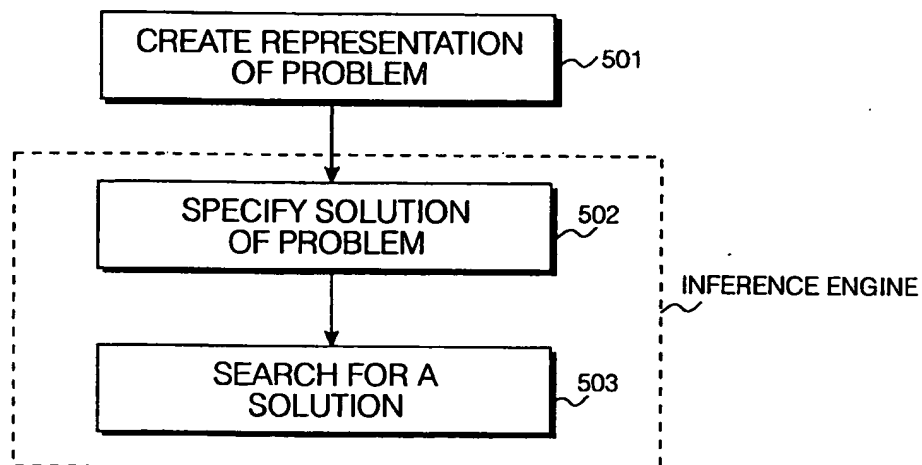


Fig. 5

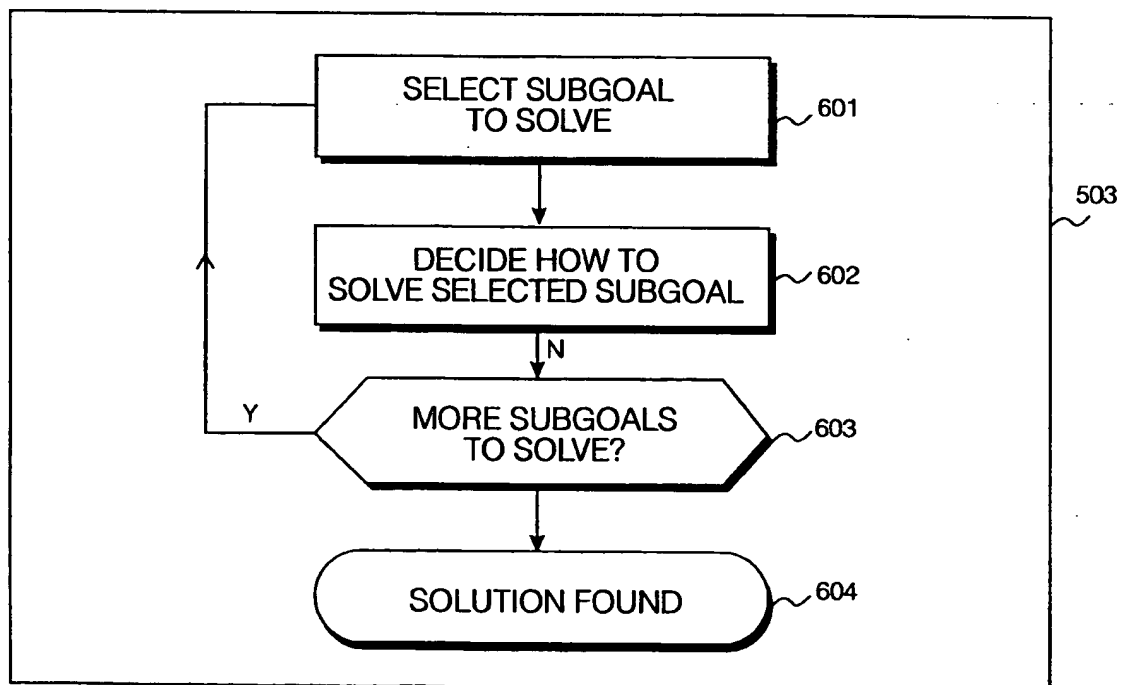


Fig. 6

4/10

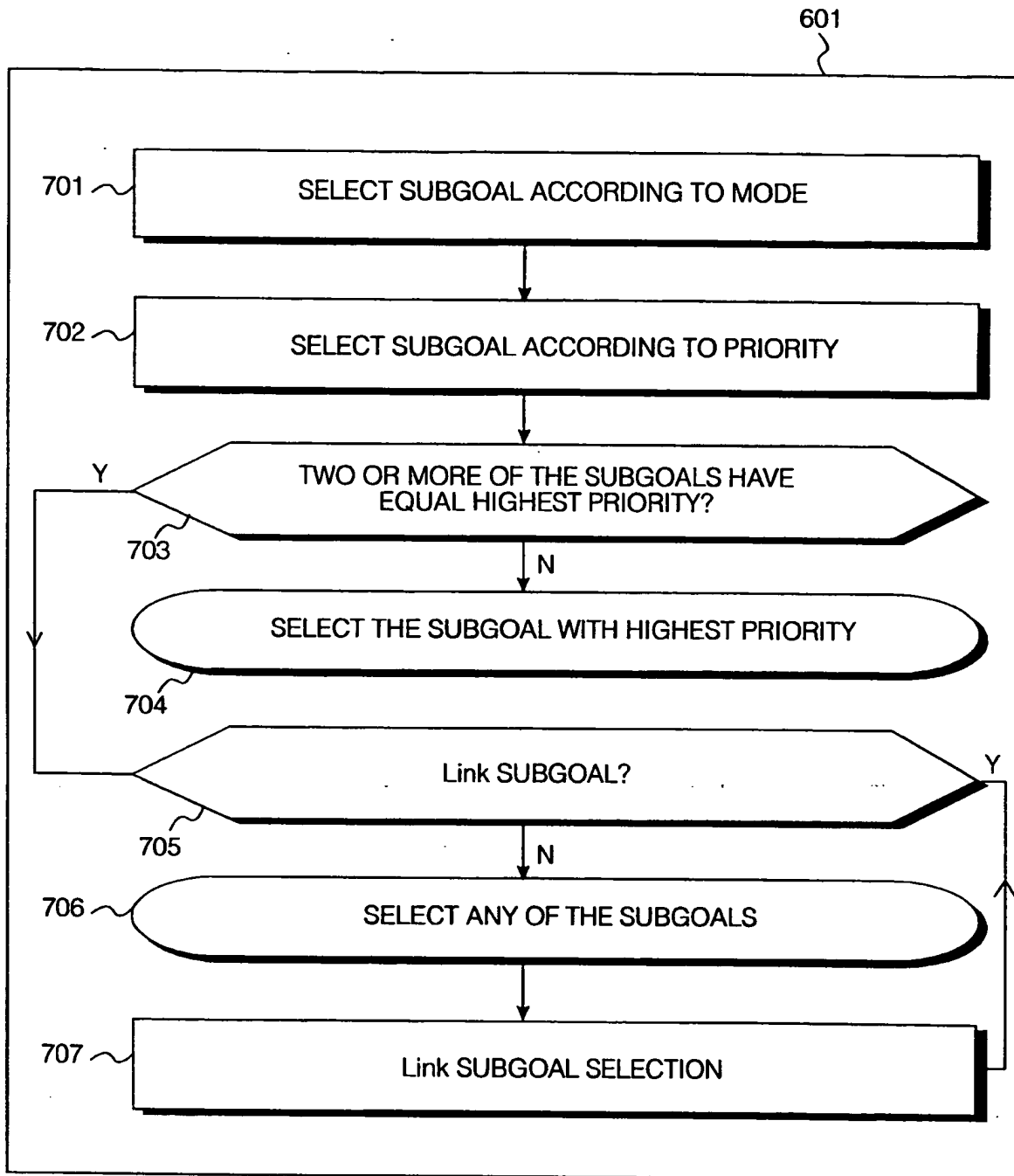


Fig. 7

5/10

Priority	Subgoal and mode
4	provision_instance(<u>T1</u> , <u>T2</u> , B, IM)
4	behavior_rule(B, Bs)
4	behavior_rule_generator(B, <u>L</u>)
3	link(<u>T1</u> , T2, <u>Link</u>)
3	link(T1, <u>T2</u> , <u>Link</u>)
3	link(<u>T1</u> , <u>T2</u> , <u>Link</u>)
2	class_info(class, Type, Level)
2	logical_link(T1, T2, Link, IM)
2	provision_endpoints(<u>L</u> , IM)
2	provision_endpoint(<u>T</u> , IM)
2	realise_endpoint(<u>Type</u> , <u>I</u> , IM)
2	provision_behaviour(<u>Type</u> , Bs, T1, T2, IM)
2	construct_sequence(Bs, <u>T1</u> , <u>T2</u> , IM)
2	add_logical_link(<u>T1</u> , <u>T2</u> , <u>Relation</u> , IM)
2	X=Y
2	<u>X=Y</u>
1	integer(Id)

Fig. 8

6/10

707

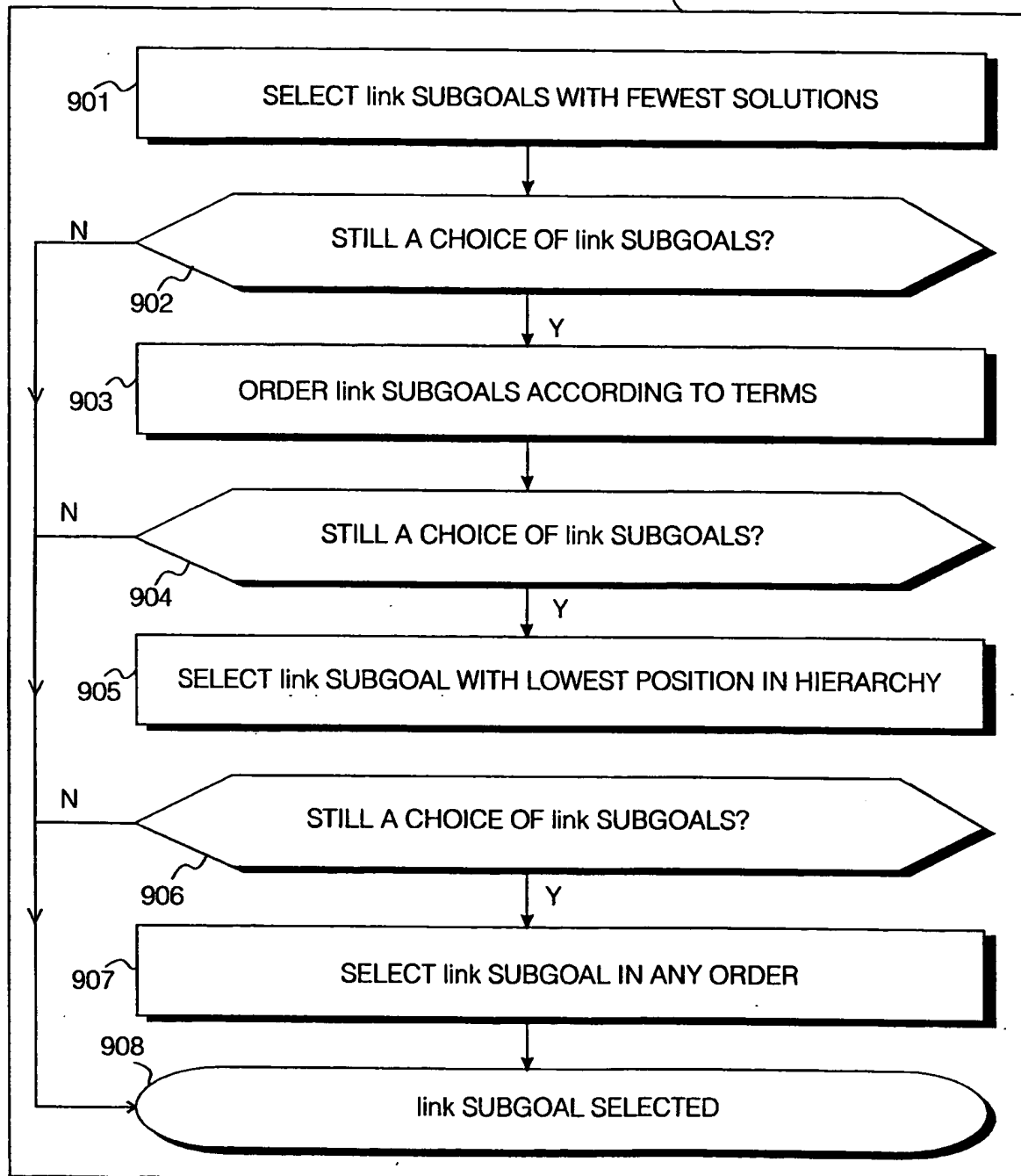


Fig. 9

7/10

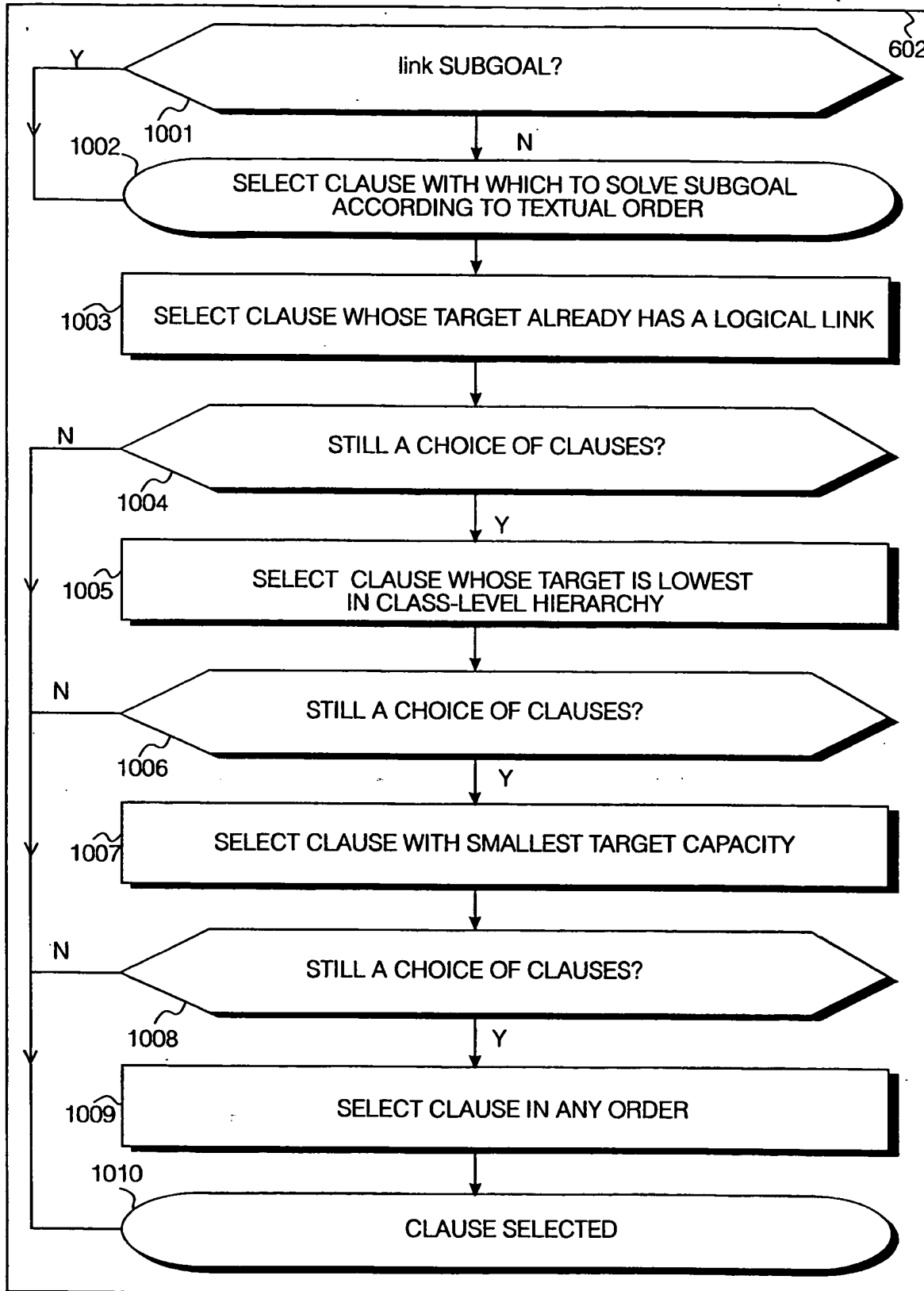


Fig. 10

9/10

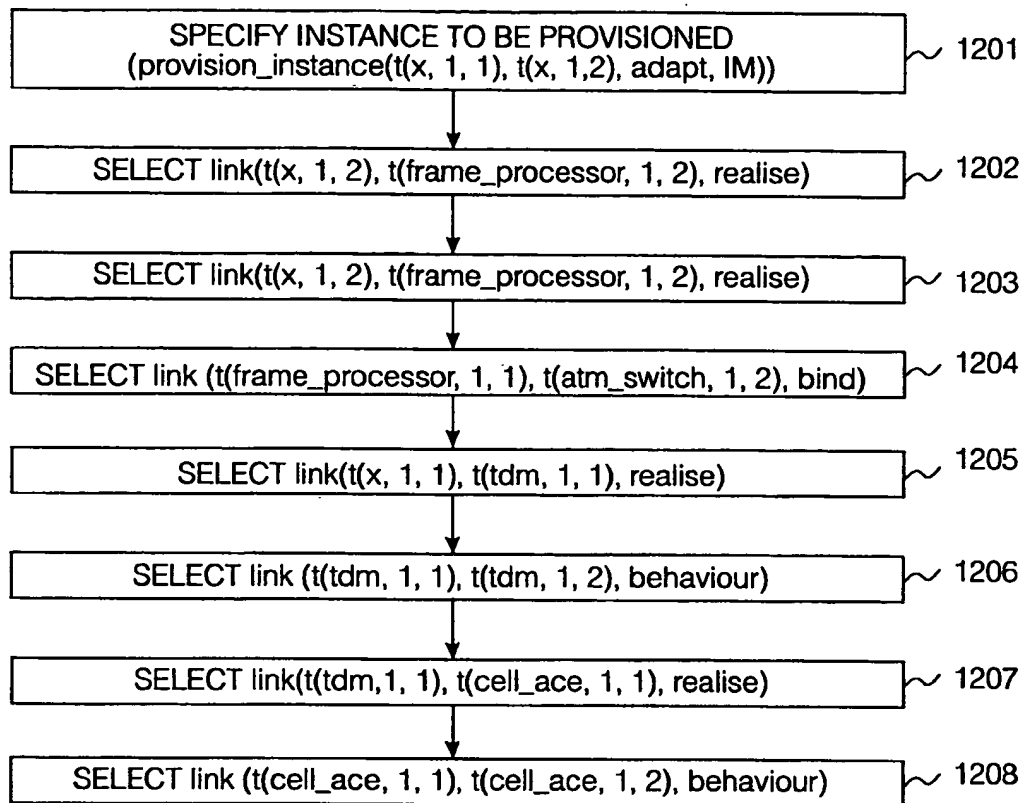


Fig. 12

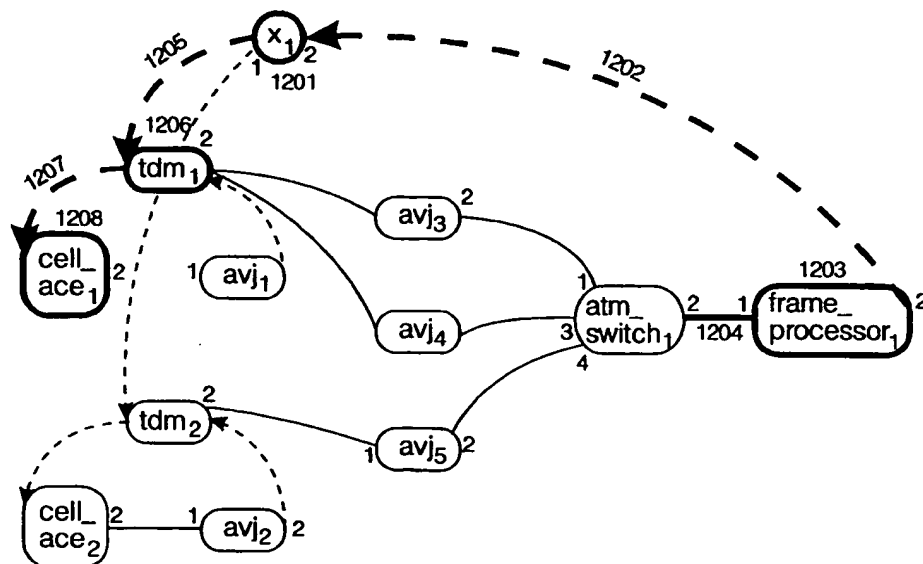


Fig. 13

10/10

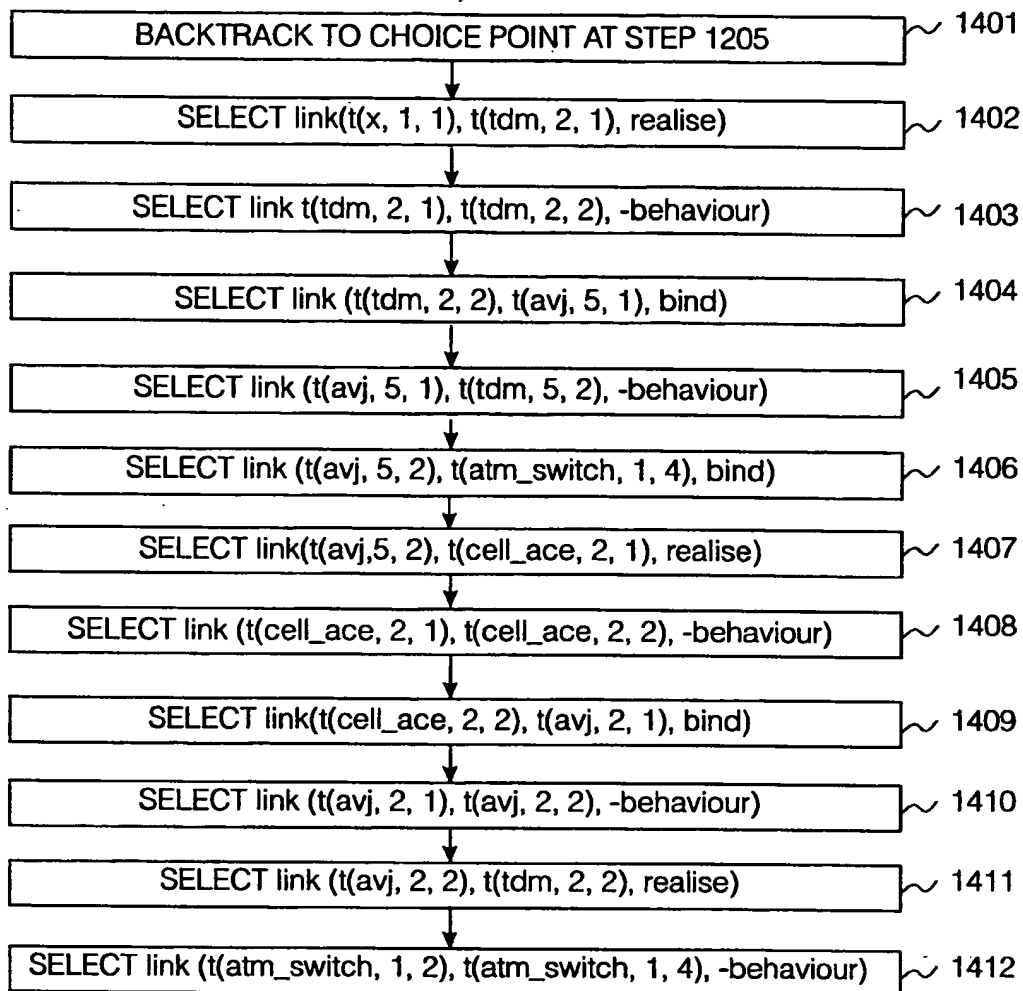


Fig. 14

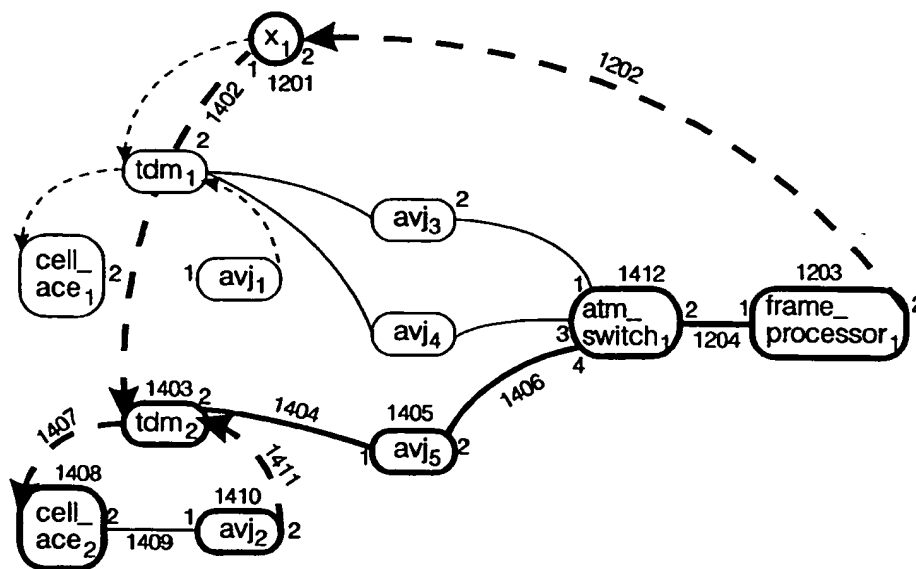


Fig. 15

INTERNATIONAL SEARCH REPORT

International Application No

PCT/GB 98/00923

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04Q11/04 H04L12/24

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC 6 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 4 999 833 A (LEE WILLIAM C) 12 March 1991	1-3, 18, 20, 21, 37, 43, 59, 63, 64
Y	see column 5, line 32 - column 6, line 32; figure 3 see column 11, line 44 - line 46	19, 38
X	EP 0 748 142 A (PLESSEY TELECOMM) 11 December 1996 see column 2, line 46 - line 58 see column 4, line 9 - column 5, line 56	1, 20
X	WO 95 35611 A (ERICSSON TELEFON AB L M ;ANDERSSON ERIK STAFFAN (SE); LINDBERG TOR) 28 December 1995 see page 3, line 28 - page 5, line 20 see page 13, line 21 - line 33	46-53
-/--		

☒ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search

22 July 1998

Date of mailing of the international search report

03/08/1998

Name and mailing address of the ISA

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,
Fax: (+31-70) 340-3016

Authorized officer

Gregori, S

INTERNATIONAL SEARCH REPORT

International Application No

PCT/GB 98/00923

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	EP 0 733 967 A (HEWLETT PACKARD CO) 25 September 1996 see page 1, line 51 - line 53 see page 6, line 14 - line 18; figure 3	19,38
A	LOH P K K: "ARTIFICIAL INTELLIGENCE SEARCH TECHNIQUES AS FAULTS-TOLERANT ROUTING STRATEGIES" PARALLEL COMPUTING, vol. 22, no. 8, 30 October 1996, pages 1127-1147, XP000635524 * sections 6-11 *	1,20,59

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No

PCT/GB 98/00923

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 4999833 A	12-03-1991	EP 0201308 A	12-11-1986
EP 0748142 A	11-12-1996	AU 5582196 A	19-12-1996
		CA 2178555 A	11-12-1996
		GB 2302235 A	08-01-1997
WO 9535611 A	28-12-1995	AU 692271 B	04-06-1998
		AU 2758595 A	15-01-1996
		CA 2191424 A	14-12-1995
		EP 0765582 A	02-04-1997
		JP 10501667 T	10-02-1998
EP 0733967 A	25-09-1996	EP 0733966 A	25-09-1996
		JP 9016537 A	17-01-1997